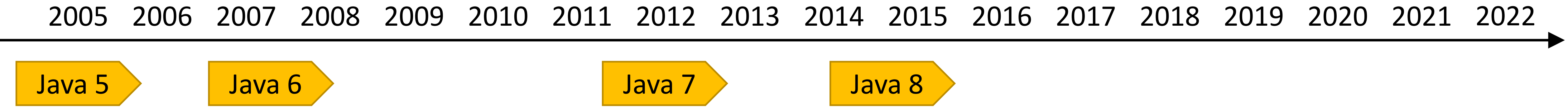
Java 17
Для тех, кто в танке

Тагир Валеев

Java release train



2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022

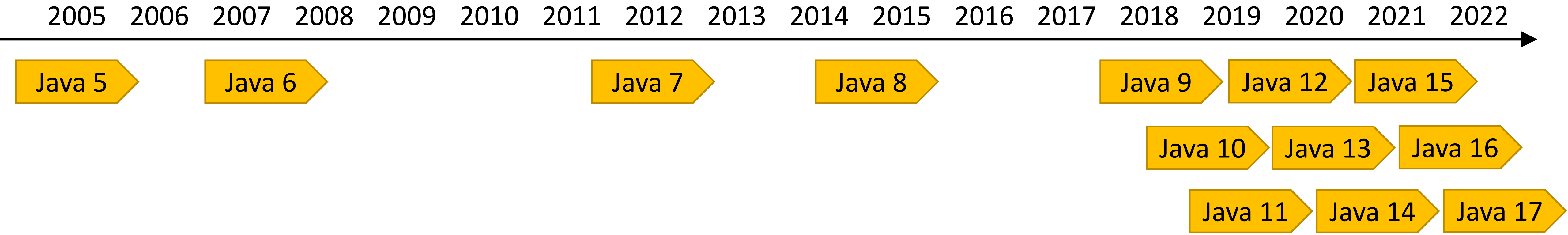
Java 5

Java 6

Java 7

Java 8

Java release train



2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022

Java 5

Java 6

Java 7

Java 8

Java 9

Java 10

Java 11

Java 12

Java 13

Java 14

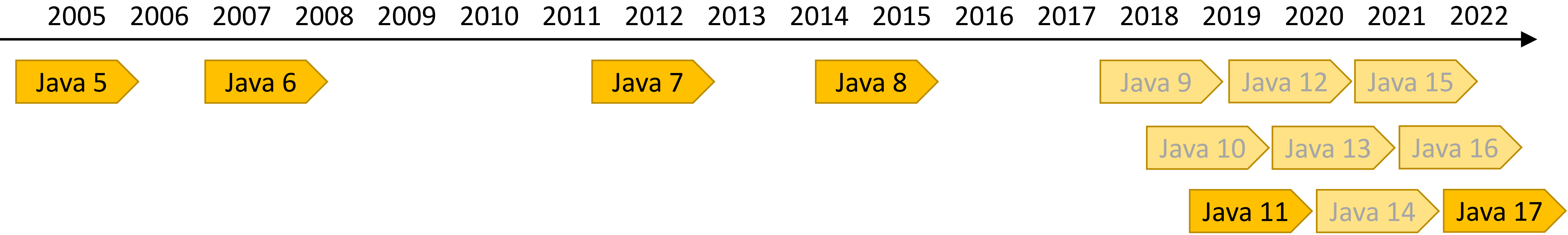
Java 15

Java 16

Java 17

Куда вы все прёте???

Java release train - LTS



2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022

Java 5

Java 6

Java 7

Java 8

Java 9

Java 12

Java 15

Java 10

Java 13

Java 16

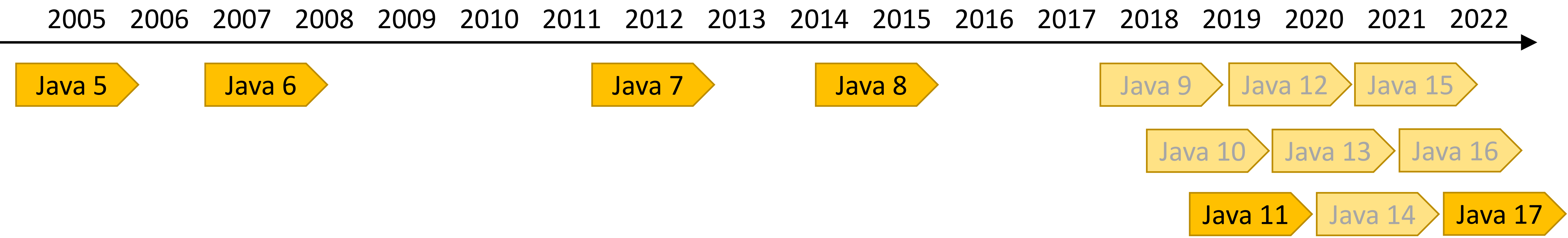
Java 11

Java 14

Java 17

Фуф, так как-то спокойнее

Java release train - LTS

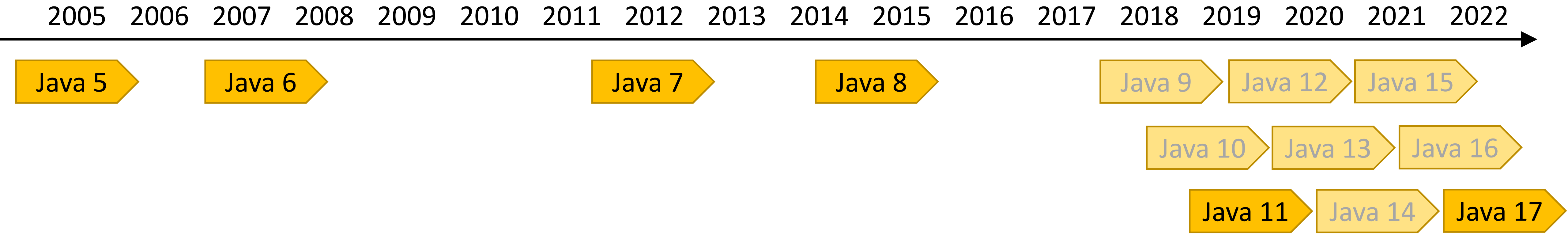


```java
boolean isLTS(int version) {
  return version <= 8 || (version - 11) % 6 == 0;
}
```

Фуф, так как-то спокойнее

Java release train - LTS

https://mreinhold.org/blog/forward-even-faster



```
boolean isLTS(int version) {
  return version <= 8 ||
         version <= 17 && (version - 11) % 6 == 0 ||
         (version - 17) % 4 == 0;
}
```

Что-то мы медленно едем…

What’s new in Java?

Крутые оптимизации производительности

Крутые фичи языка Java

Крутые улучшения безопасности

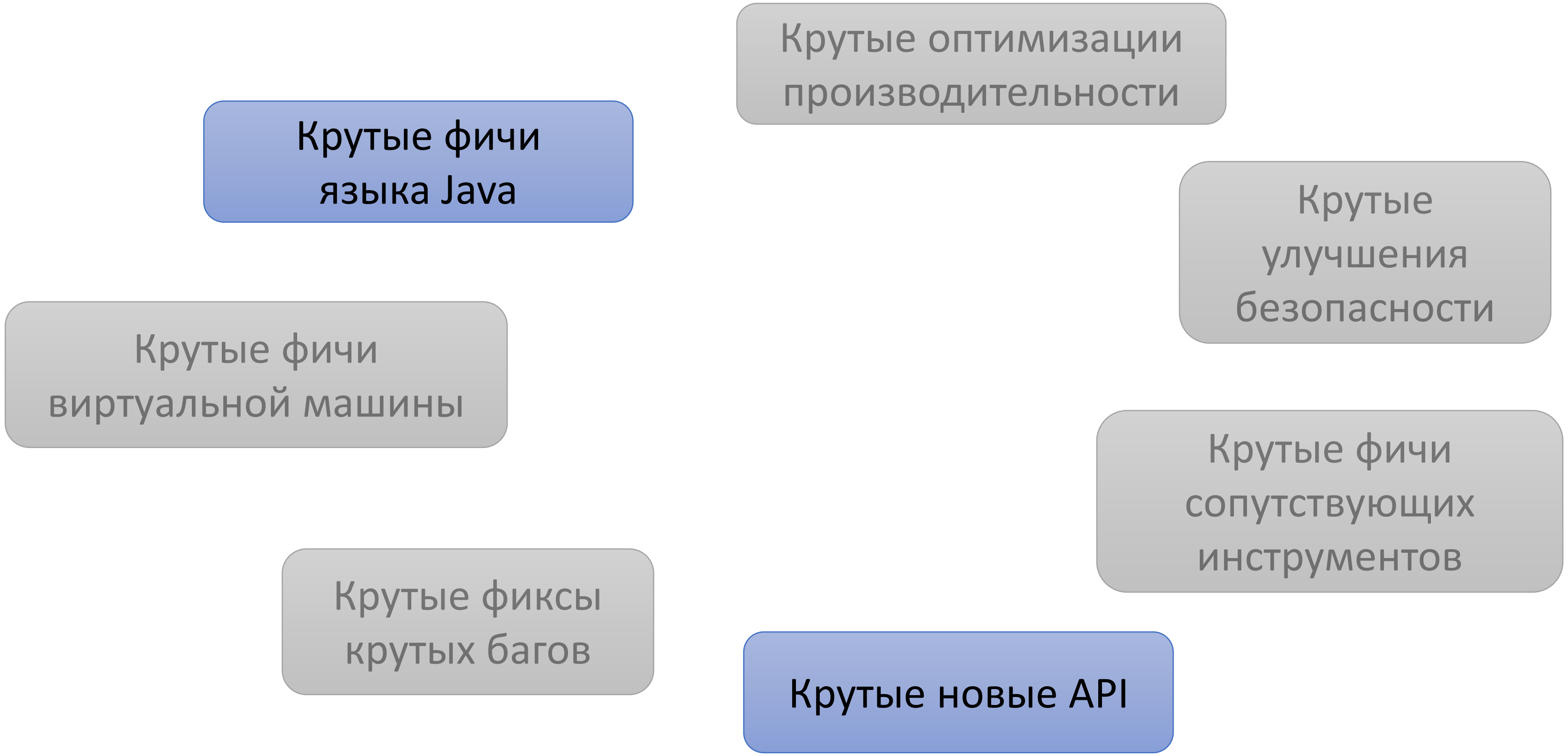
Крутые фичи виртуальной машины

Крутые фичи сопутствующих инструментов

Крутые фиксы крутых багов

Крутые новые API

What’s new in Java?



Крутые оптимизации производительности

Крутые фичи языка Java

Крутые улучшения безопасности

Крутые фичи виртуальной машины

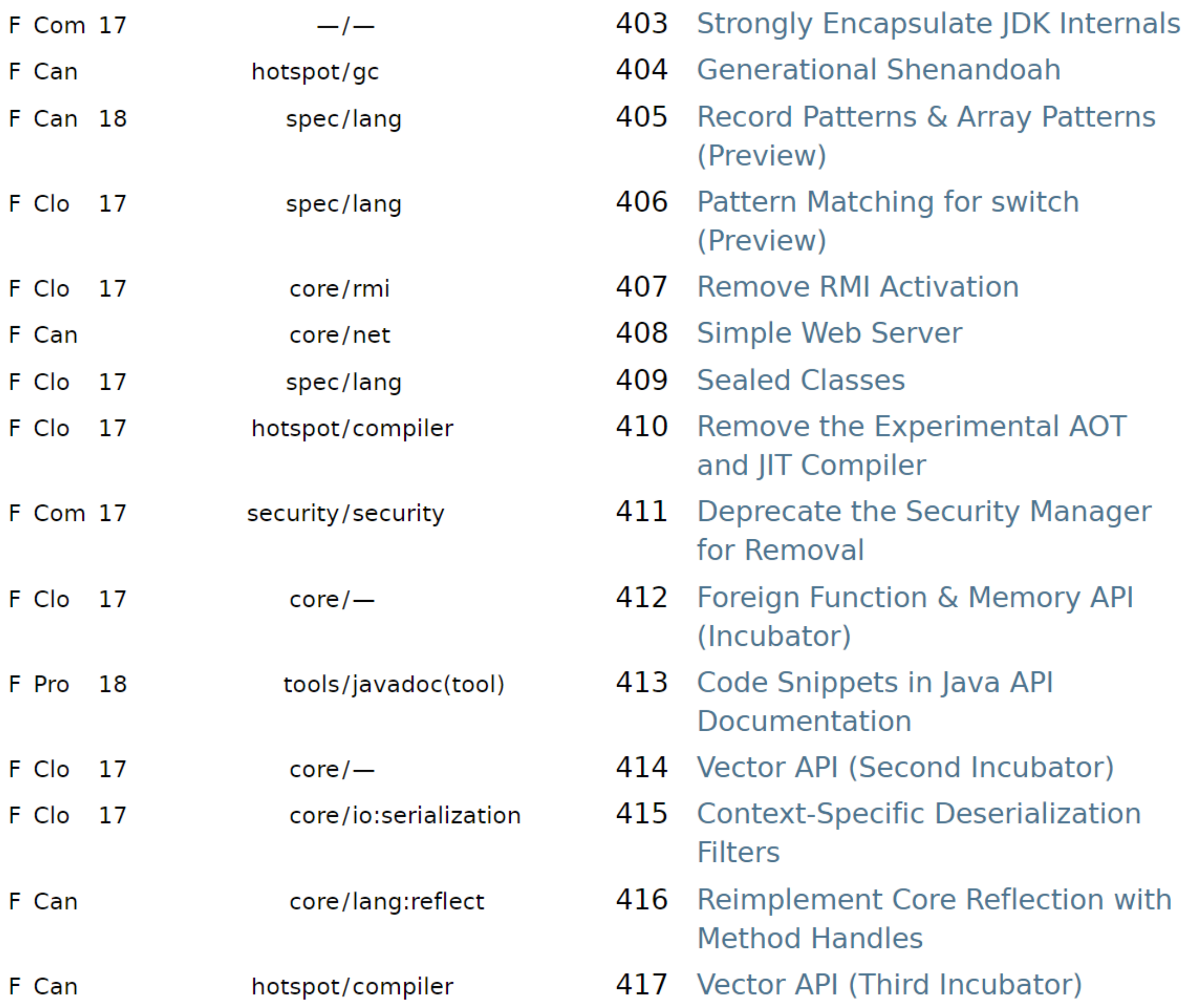
Крутые фичи сопутствующих инструментов

Крутые фиксы крутых багов

Крутые новые API

JEP = JDK Enhancement Proposals

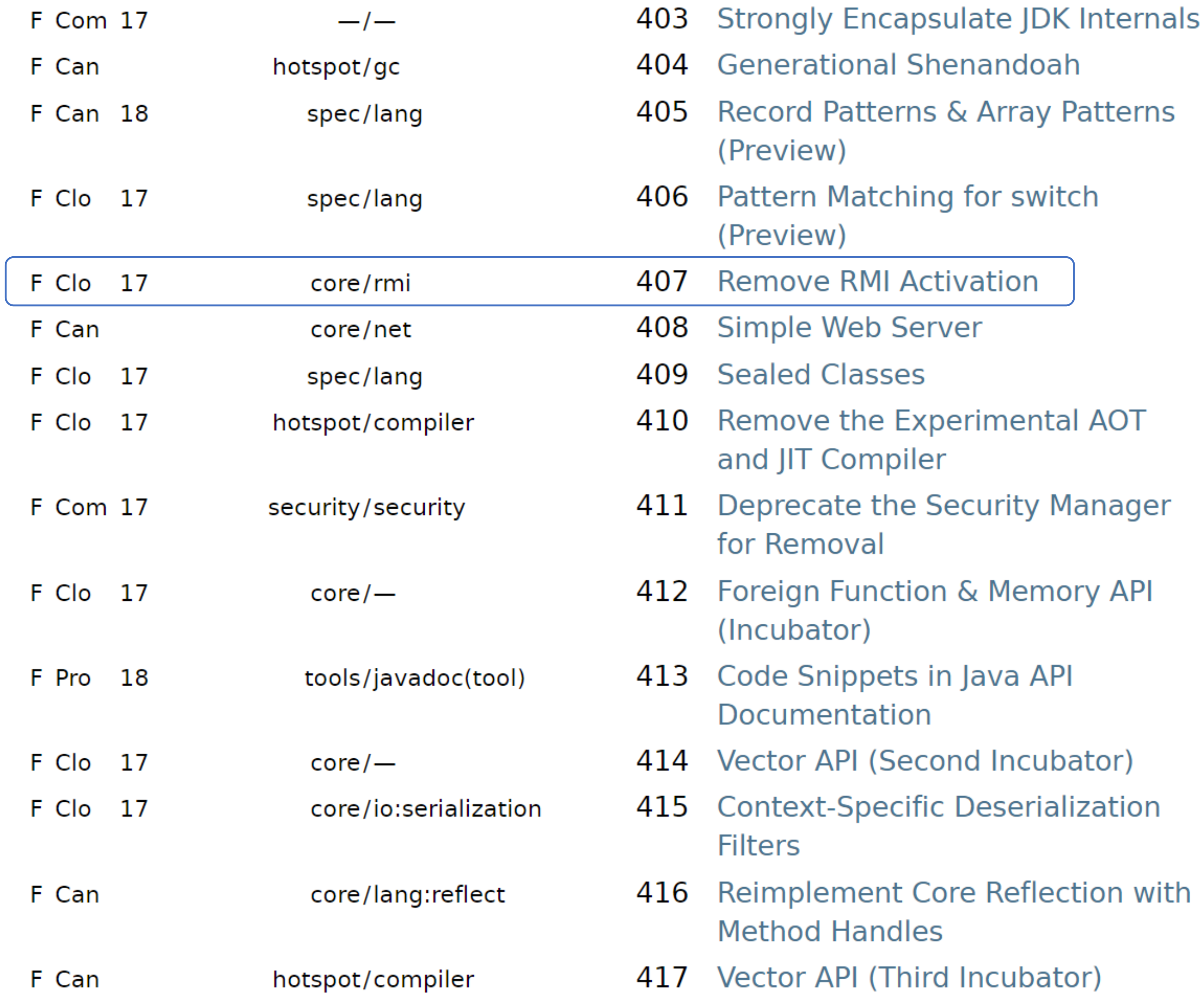
https://openjdk.java.net/jeps/0 — список



F	Com	17	—/—	403	Strongly Encapsulate JDK Internals
F	Can		hotspot/gc	404	Generational Shenandoah
F	Can	18	spec/lang	405	Record Patterns & Array Patterns (Preview)
F	Clo	17	spec/lang	406	Pattern Matching for switch (Preview)
F	Clo	17	core/rmi	407	Remove RMI Activation
F	Can		core/net	408	Simple Web Server
F	Clo	17	spec/lang	409	Sealed Classes
F	Clo	17	hotspot/compiler	410	Remove the Experimental AOT and JIT Compiler
F	Com	17	security/security	411	Deprecate the Security Manager for Removal
F	Clo	17	core/—	412	Foreign Function & Memory API (Incubator)
F	Pro	18	tools/javadoc(tool)	413	Code Snippets in Java API Documentation
F	Clo	17	core/—	414	Vector API (Second Incubator)
F	Clo	17	core/io:serialization	415	Context-Specific Deserialization Filters
F	Can		core/lang:reflect	416	Reimplement Core Reflection with Method Handles
F	Can		hotspot/compiler	417	Vector API (Third Incubator)

JEP = JDK Enhancement Proposals

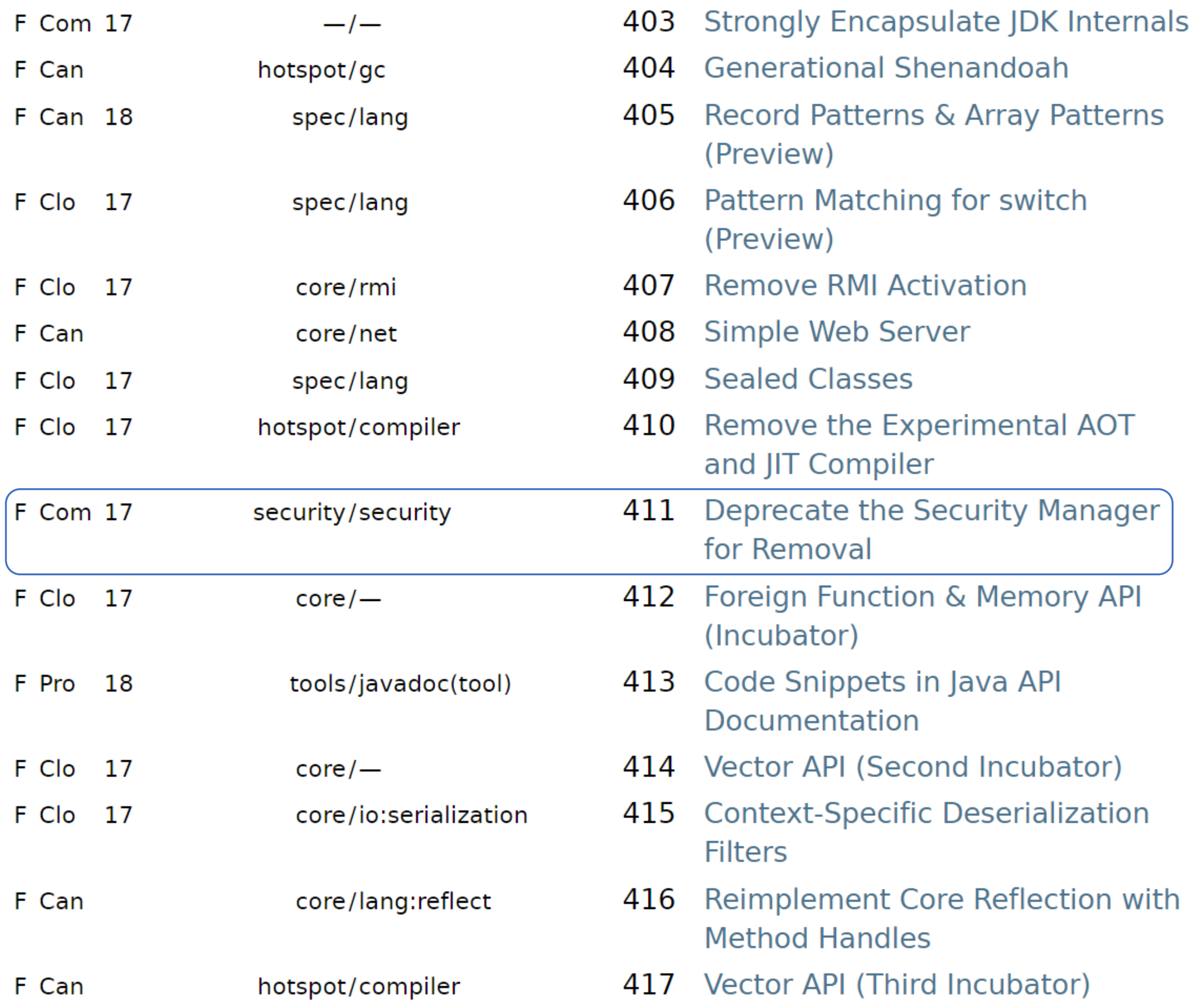
https://openjdk.java.net/jeps/0 — список



F	Com	17	—/—	403	Strongly Encapsulate JDK Internals
F	Can		hotspot/gc	404	Generational Shenandoah
F	Can	18	spec/lang	405	Record Patterns & Array Patterns (Preview)
F	Clo	17	spec/lang	406	Pattern Matching for switch (Preview)
F	Clo	17	core/rmi	407	Remove RMI Activation
F	Can		core/net	408	Simple Web Server
F	Clo	17	spec/lang	409	Sealed Classes
F	Clo	17	hotspot/compiler	410	Remove the Experimental AOT and JIT Compiler
F	Com	17	security/security	411	Deprecate the Security Manager for Removal
F	Clo	17	core/—	412	Foreign Function & Memory API (Incubator)
F	Pro	18	tools/javadoc(tool)	413	Code Snippets in Java API Documentation
F	Clo	17	core/—	414	Vector API (Second Incubator)
F	Clo	17	core/io:serialization	415	Context-Specific Deserialization Filters
F	Can		core/lang:reflect	416	Reimplement Core Reflection with Method Handles
F	Can		hotspot/compiler	417	Vector API (Third Incubator)

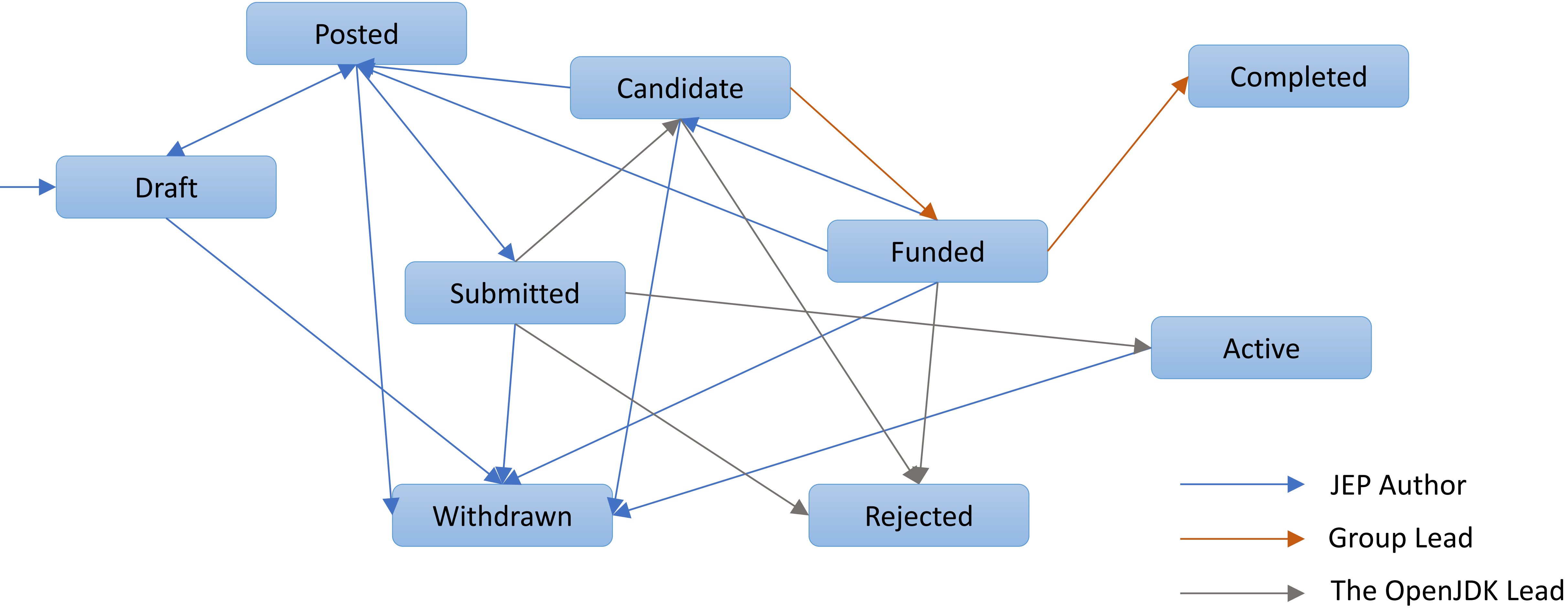
JEP = JDK Enhancement Proposals

https://openjdk.java.net/jeps/0 — список



F	Com	17	—/—	403	Strongly Encapsulate JDK Internals
F	Can		hotspot/gc	404	Generational Shenandoah
F	Can	18	spec/lang	405	Record Patterns & Array Patterns (Preview)
F	Clo	17	spec/lang	406	Pattern Matching for switch (Preview)
F	Clo	17	core/rmi	407	Remove RMI Activation
F	Can		core/net	408	Simple Web Server
F	Clo	17	spec/lang	409	Sealed Classes
F	Clo	17	hotspot/compiler	410	Remove the Experimental AOT and JIT Compiler
F	Com	17	security/security	411	Deprecate the Security Manager for Removal
F	Clo	17	core/—	412	Foreign Function & Memory API (Incubator)
F	Pro	18	tools/javadoc(tool)	413	Code Snippets in Java API Documentation
F	Clo	17	core/—	414	Vector API (Second Incubator)
F	Clo	17	core/io:serialization	415	Context-Specific Deserialization Filters
F	Can		core/lang:reflect	416	Reimplement Core Reflection with Method Handles
F	Can		hotspot/compiler	417	Vector API (Third Incubator)

JEP 1: JDK Enhancement-Proposal & Roadmap Process



https://openjdk.java.net/jeps/1

JEP 12: Preview Features

```
$ javac --release 17 --enable-preview Foo.java
Note: Some input files use a preview language feature.
Note: Recompile with -Xlint:preview for details.

$ java --enable-preview Foo
```

https://openjdk.java.net/jeps/12

JEP 12: Preview Features

```
$ javac --release 17 --enable-preview Foo.java
Note: Some input files use a preview language feature.
Note: Recompile with -Xlint:preview for details.

$ java --enable-preview Foo
```

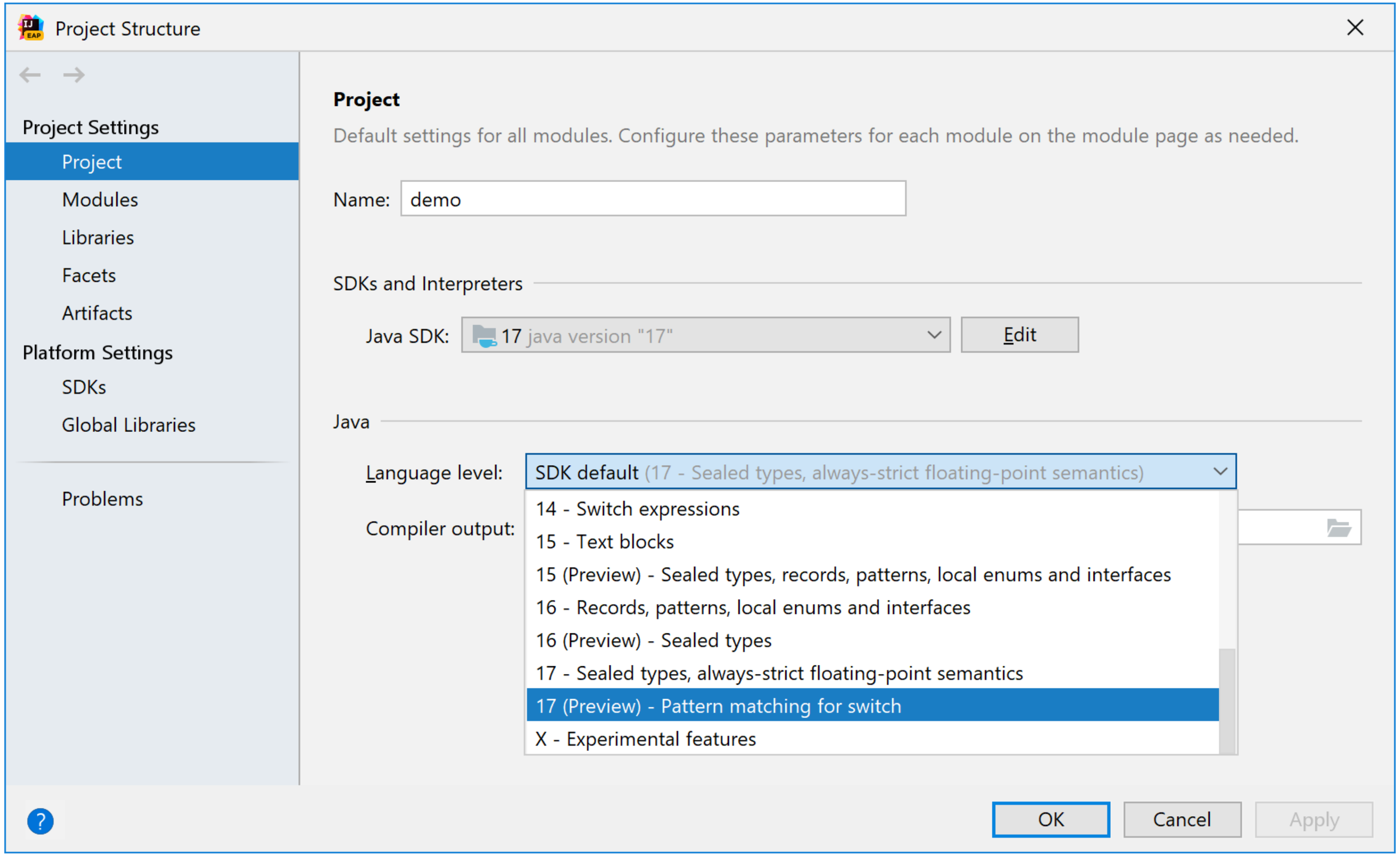
https://openjdk.java.net/jeps/12

JEP 12: Preview Features

```
$ javac --release 17 --enable-preview Foo.java
Note: Some input files use a preview language feature.
Note: Recompile with -Xlint:preview for details.

$ java --enable-preview Foo
```

https://openjdk.java.net/jeps/12

JEP 12: Preview Features



Project

Default settings for all modules. Configure these parameters for each module on the module page as needed.

Name: demo

SDKs and Interpreters

Java SDK: 17 java version "17"

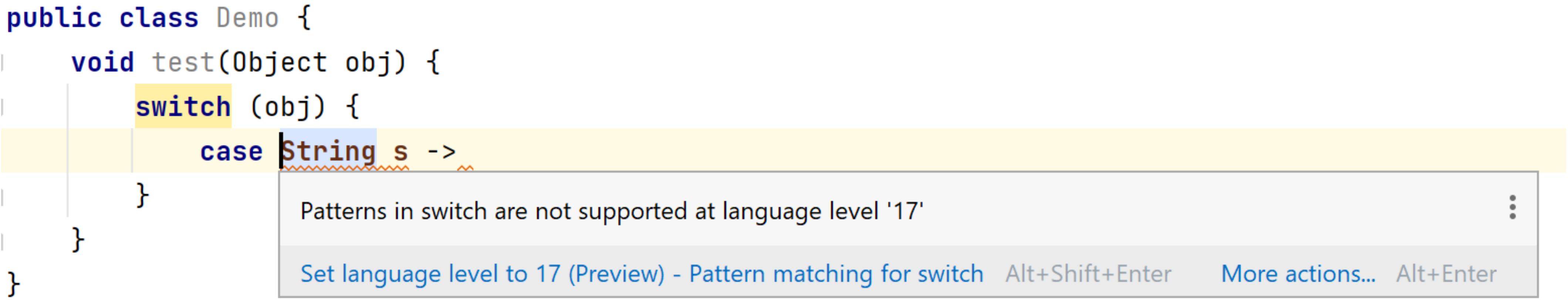
Java

Language level: SDK default (17 - Sealed types, always-strict floating-point semantics)

- 14 - Switch expressions
- 15 - Text blocks
- 15 (Preview) - Sealed types, records, patterns, local enums and interfaces
- 16 - Records, patterns, local enums and interfaces
- 16 (Preview) - Sealed types
- 17 - Sealed types, always-strict floating-point semantics
- 17 (Preview) - Pattern matching for switch
- X - Experimental features

Compiler output:

JEP 12: Preview Features

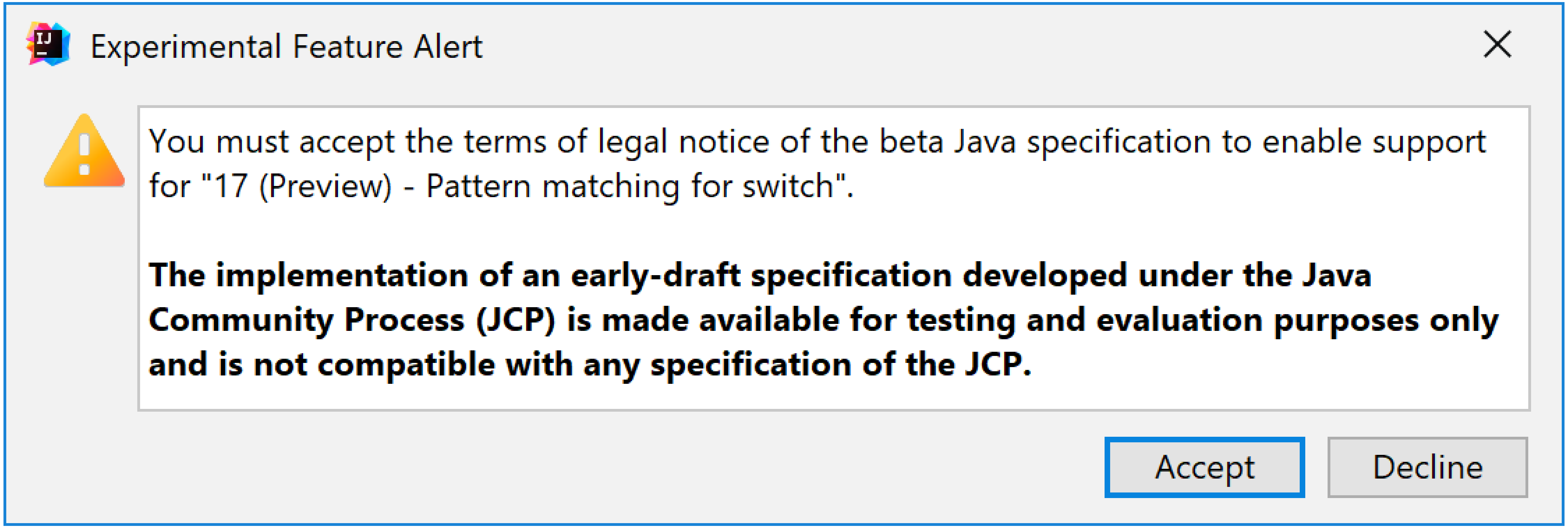


```java
public class Demo {
    void test(Object obj) {
        switch (obj) {
            case String s ->
        }
    }
}
```

Patterns in switch are not supported at language level '17'

Set language level to 17 (Preview) - Pattern matching for switch
Alt+Shift+Enter
More actions...
Alt+Enter

JEP 12: Preview Features

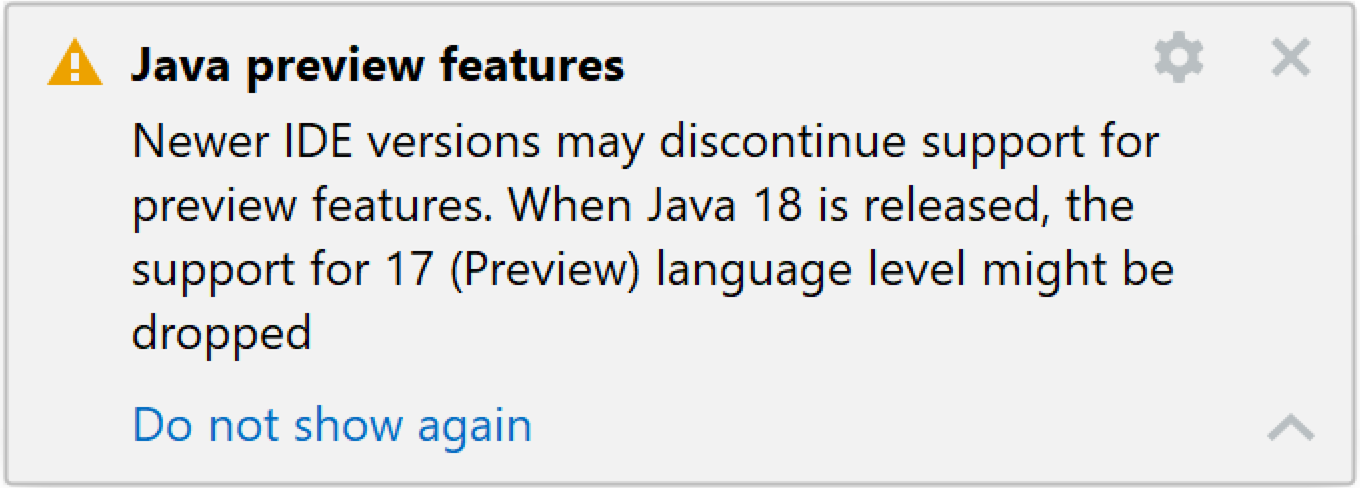


Experimental Feature Alert

You must accept the terms of legal notice of the beta Java specification to enable support for "17 (Preview) - Pattern matching for switch".

The implementation of an early-draft specification developed under the Java Community Process (JCP) is made available for testing and evaluation purposes only and is not compatible with any specification of the JCP.

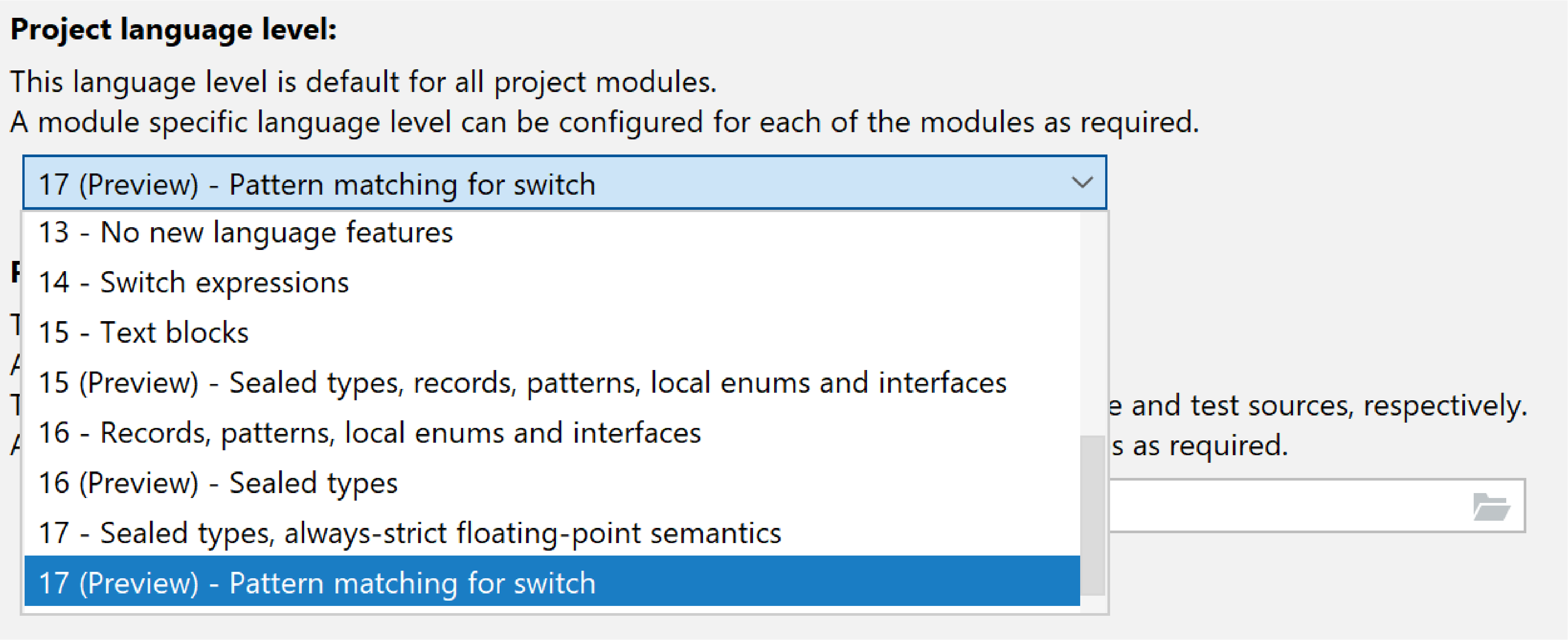
JEP 12: Preview Features



Java preview features

Newer IDE versions may discontinue support for preview features. When Java 18 is released, the support for 17 (Preview) language level might be dropped

JEP 12: Preview Features



Project language level:

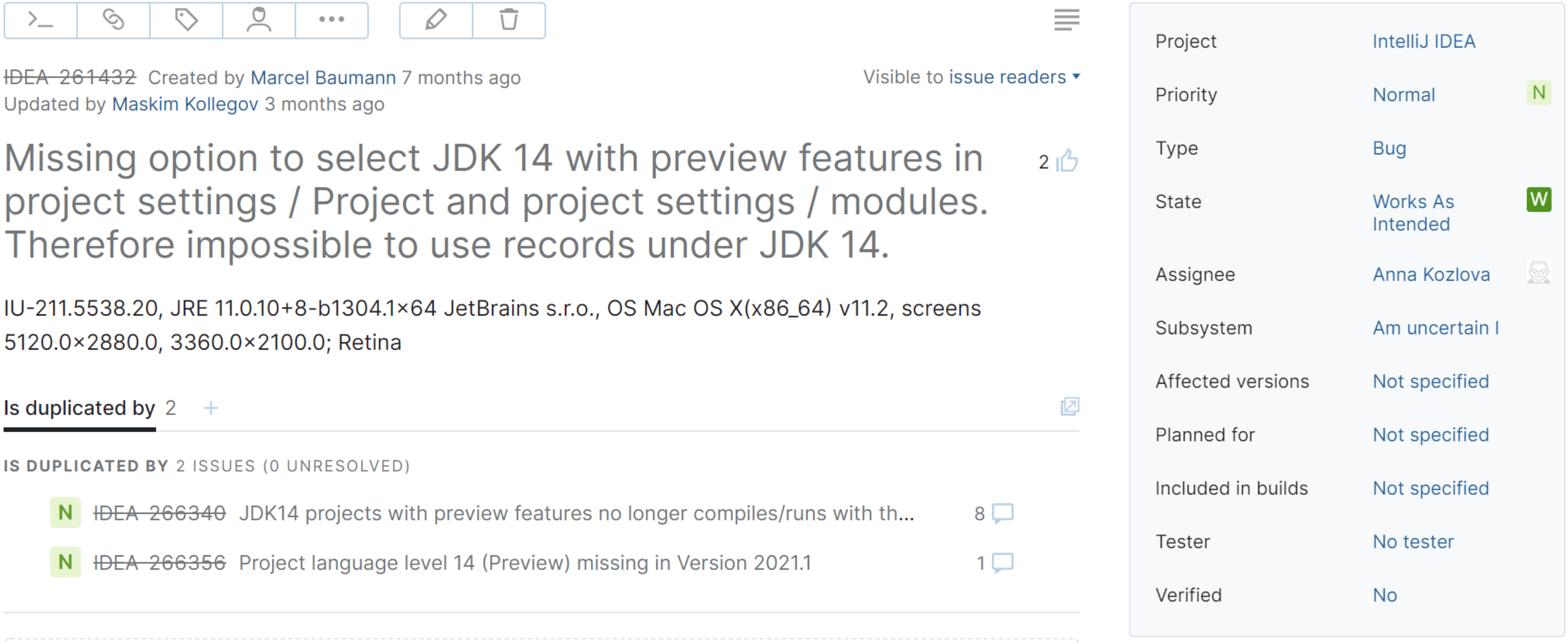
This language level is default for all project modules.
A module specific language level can be configured for each of the modules as required.

17 (Preview) - Pattern matching for switch

- 13 - No new language features
- 14 - Switch expressions
- 15 - Text blocks
- 15 (Preview) - Sealed types, records, patterns, local enums and interfaces
- 16 - Records, patterns, local enums and interfaces
- 16 (Preview) - Sealed types
- 17 - Sealed types, always-strict floating-point semantics
- 17 (Preview) - Pattern matching for switch

e and test sources, respectively.
s as required.

JEP 12: Preview Features



~~IDEA-261432~~ Created by Marcel Baumann 7 months ago
Updated by Maskim Kollegov 3 months ago

Visible to issue readers

Missing option to select JDK 14 with preview features in project settings / Project and project settings / modules. Therefore impossible to use records under JDK 14.

2

IU-211.5538.20, JRE 11.0.10+8-b1304.1×64 JetBrains s.r.o., OS Mac OS X(x86_64) v11.2, screens 5120.0×2880.0, 3360.0×2100.0; Retina

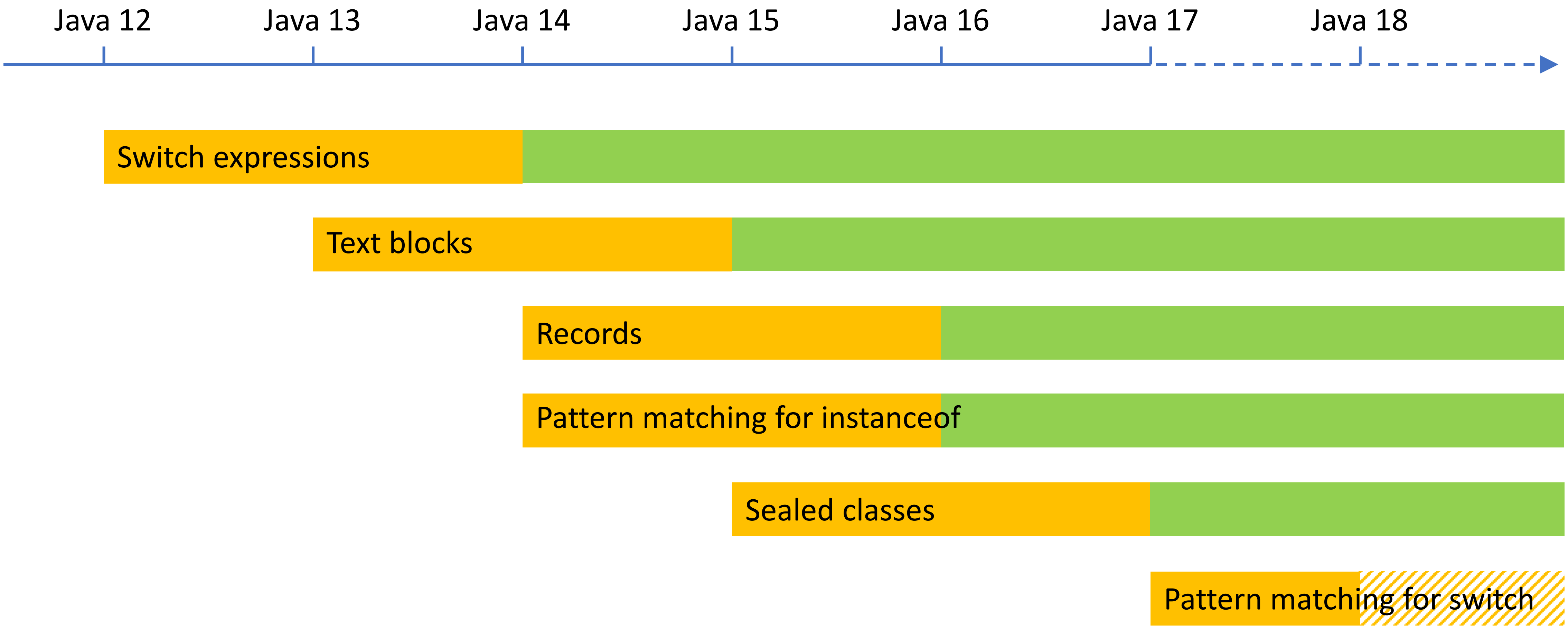
Is duplicated by 2 +

IS DUPLICATED BY 2 ISSUES (0 UNRESOLVED)

- N ~~IDEA-266340~~ JDK14 projects with preview features no longer compiles/runs with th... 8
- N ~~IDEA-266356~~ Project language level 14 (Preview) missing in Version 2021.1 1

Project	IntelliJ IDEA	
Priority	Normal	N
Type	Bug	
State	Works As Intended	W
Assignee	Anna Kozlova	
Subsystem	Am uncertain I	
Affected versions	Not specified	
Planned for	Not specified	
Included in builds	Not specified	
Tester	No tester	
Verified	No	

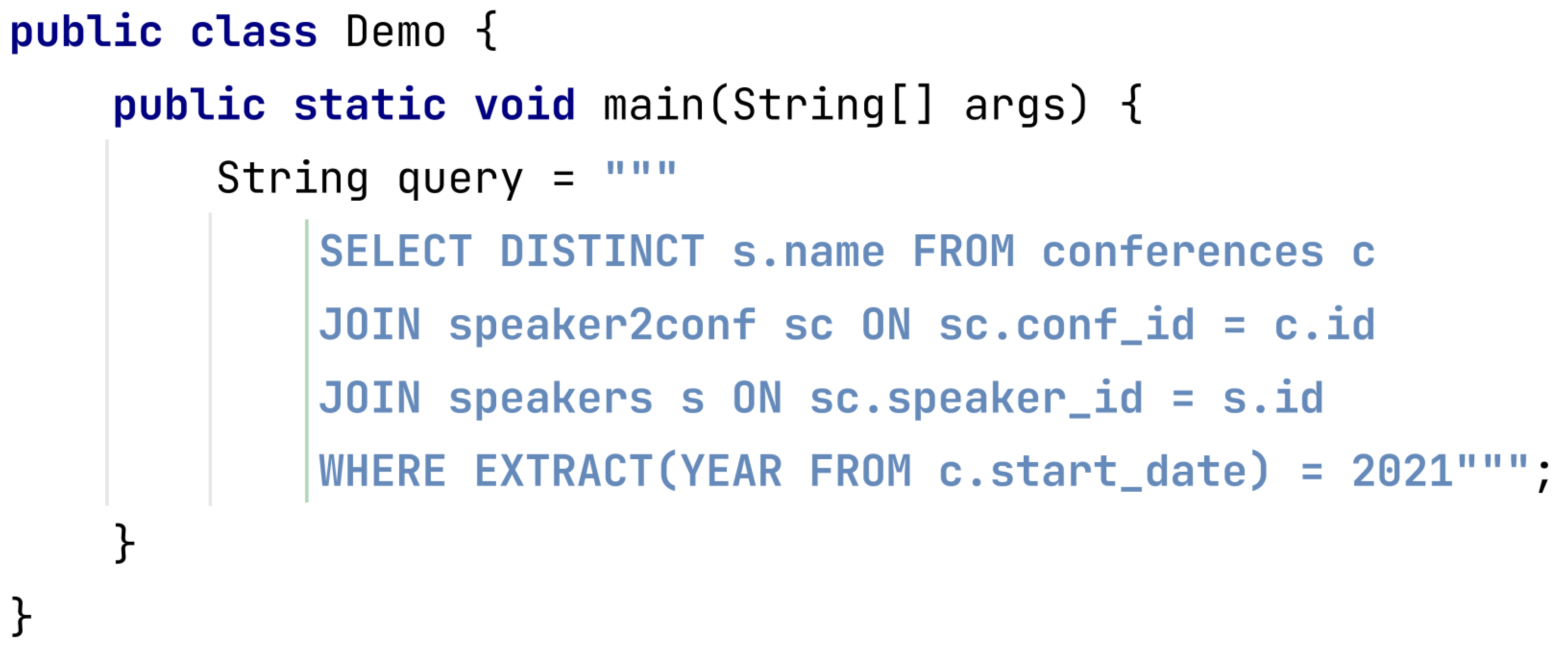
JEP 12: Preview Features



https://openjdk.java.net/jeps/12

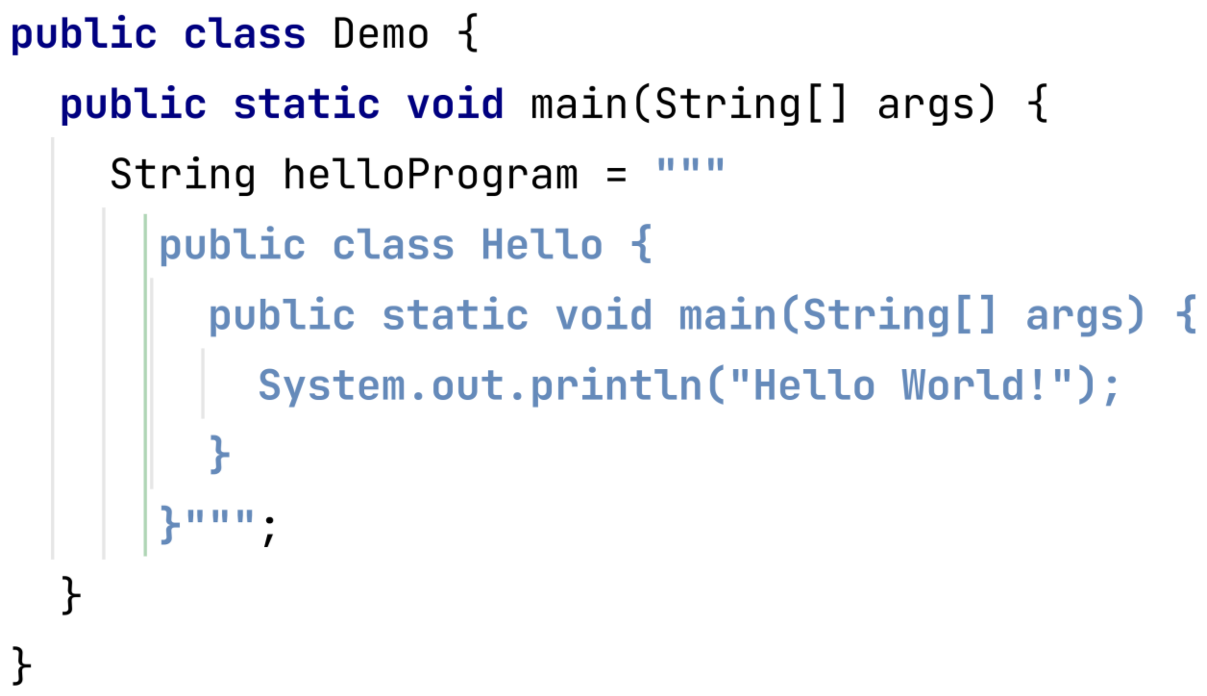
Text blocks

Java 13: Preview
Java 15: Standard



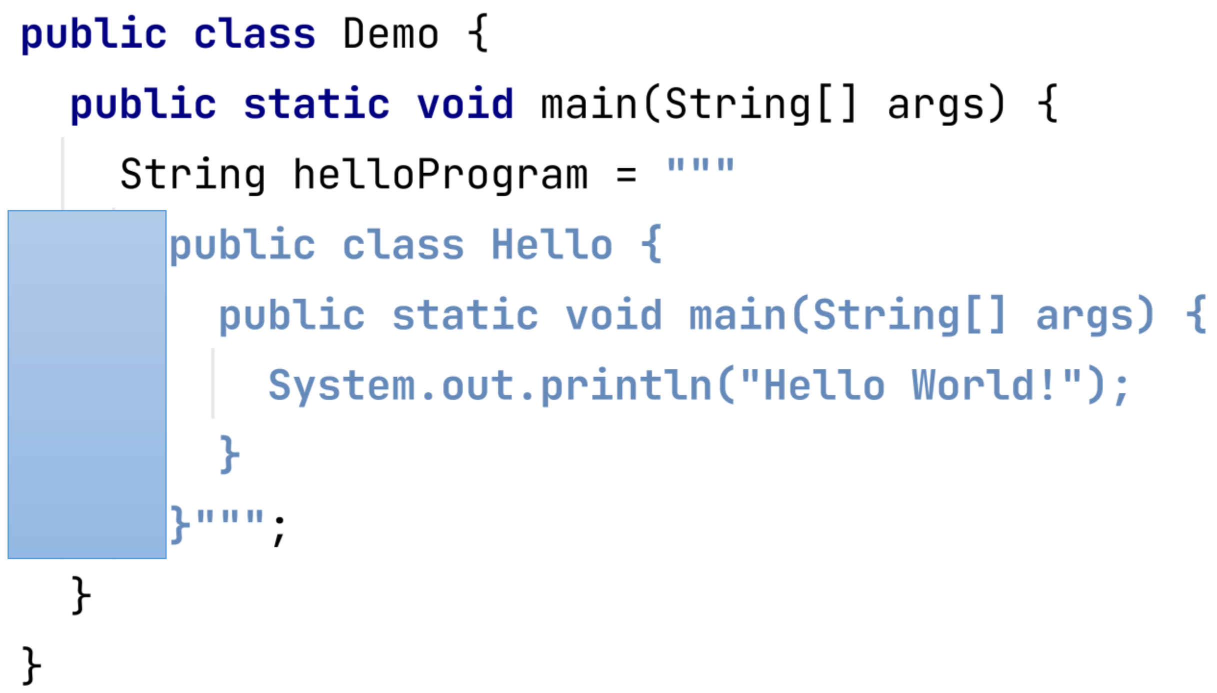
```
public class Demo {
    public static void main(String[] args) {
        String query = """
            SELECT DISTINCT s.name FROM conferences c
            JOIN speaker2conf sc ON sc.conf_id = c.id
            JOIN speakers s ON sc.speaker_id = s.id
            WHERE EXTRACT(YEAR FROM c.start_date) = 2021""";
    }
}
```

Text blocks



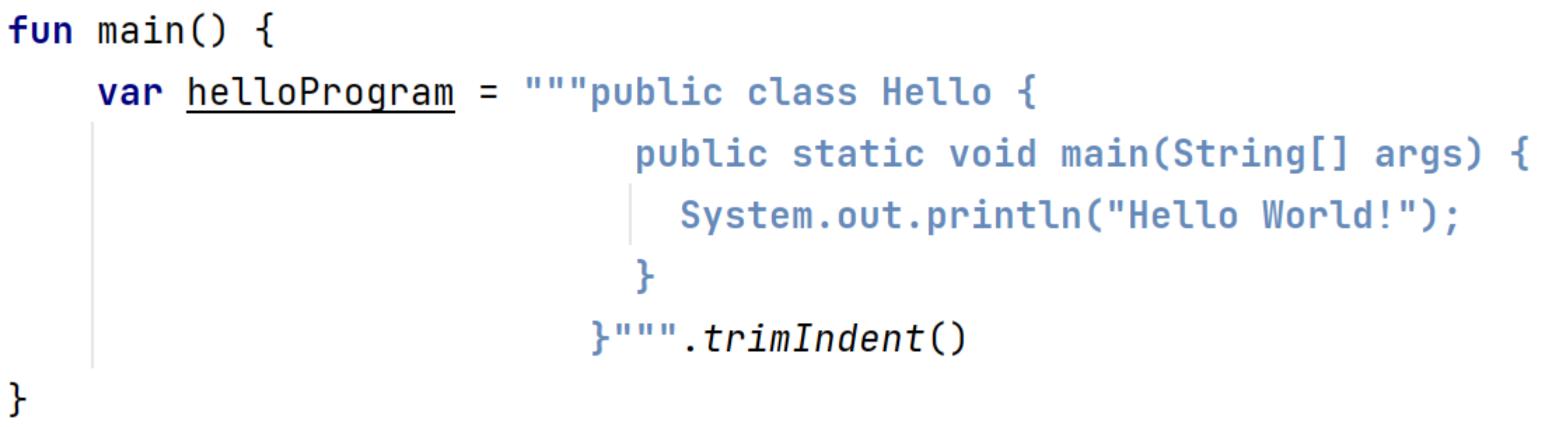
```java
public class Demo {
  public static void main(String[] args) {
    String helloProgram = """
      public class Hello {
        public static void main(String[] args) {
          System.out.println("Hello World!");
        }
      }""";
  }
}
```

Text blocks

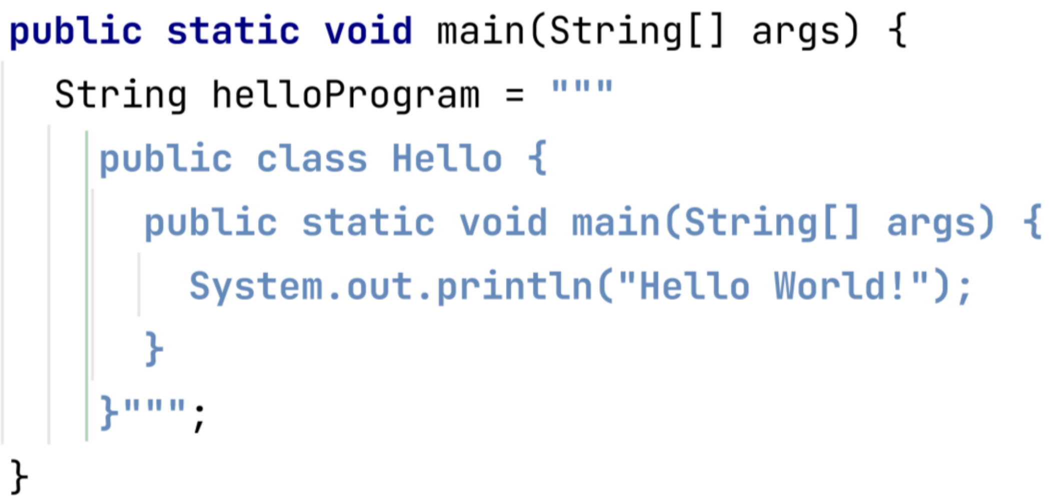


```
public class Demo {
  public static void main(String[] args) {
    String helloProgram = """
      public class Hello {
        public static void main(String[] args) {
          System.out.println("Hello World!");
        }
      }""";
  }
}
```

Text blocks

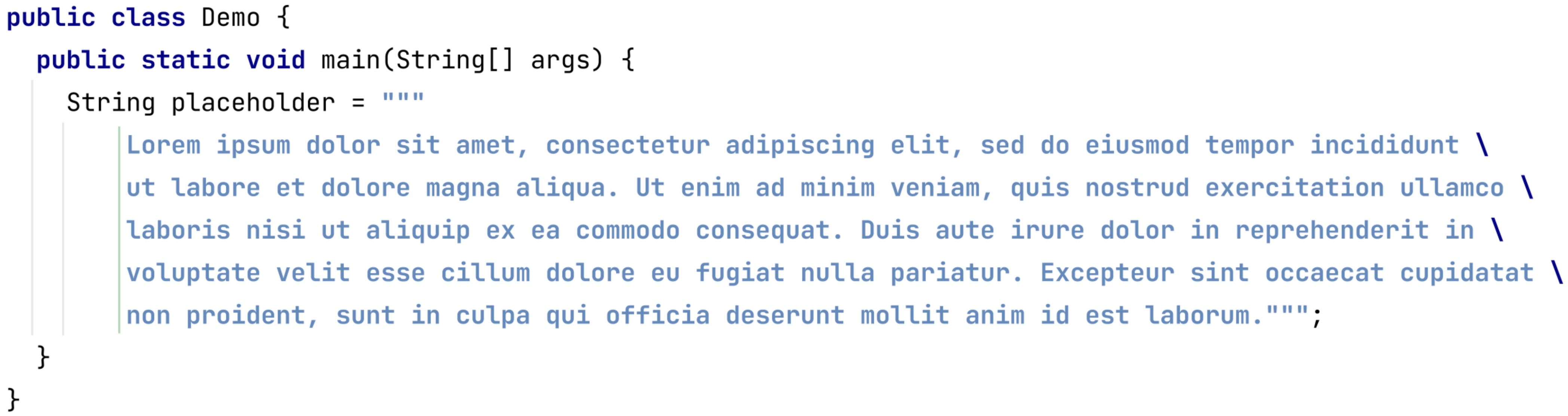


```kotlin
fun main() {
    var helloProgram = """public class Hello {
                            public static void main(String[] args) {
                              System.out.println("Hello World!");
                            }
                          }""".trimIndent()
}
```



```java
public static void main(String[] args) {
  String helloProgram = """
    public class Hello {
      public static void main(String[] args) {
        System.out.println("Hello World!");
      }
    }""";
}
```

Text blocks



```java
public class Demo {
  public static void main(String[] args) {
    String placeholder = """
        Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt \
        ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco \
        laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in \
        voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat \
        non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.""";
  }
}
```

Text blocks

```
public class StudentsDTO {
  private Connection conn;

  public void addStudent(String name, int grade) throws SQLException {
    String query = """
        INSERT INTO Students(grade, name)
        VALUES(%d, '%s')""".formatted(grade, name);
    conn.createStatement().execute(query);
  }
}
```

Text blocks

```
public class StudentsDTO {
  private Connection conn;

  public void addStudent(String name, int grade) throws SQLException {
    String query = """
        INSERT INTO Students(grade, name)
        VALUES(%d, '%s')""".formatted(grade, name);
    conn.createStatement().execute(query);
  }
}
```

```
String name = "Robert'); DROP TABLE Students;--";
int grade = 1;
addStudent(name, grade);
```

String Tapas Redux: Beyond mere string interpolation

https://github.com/openjdk/amber-docs/blob/master/site/design-notes/templated-strings.md

```
public class StudentsDTO {
  private Connection conn;

  public void addStudent(String name, int grade) throws SQLException {
    Statement statement = conn."""
        INSERT INTO Students(grade, name)
        VALUES(\{grade}, \{name})""";
    statement.execute(query);
  }
}
```

```
String name = "Robert'); DROP TABLE Students;--";
int grade = 1;
addStudent(name, grade);
```

Text blocks

```java
String myRegexp = "(\\w+)\\\\(\\w+)";
String myRegexpTextBlock = """
  (\\w+)\\\\(\\w+)""";
```

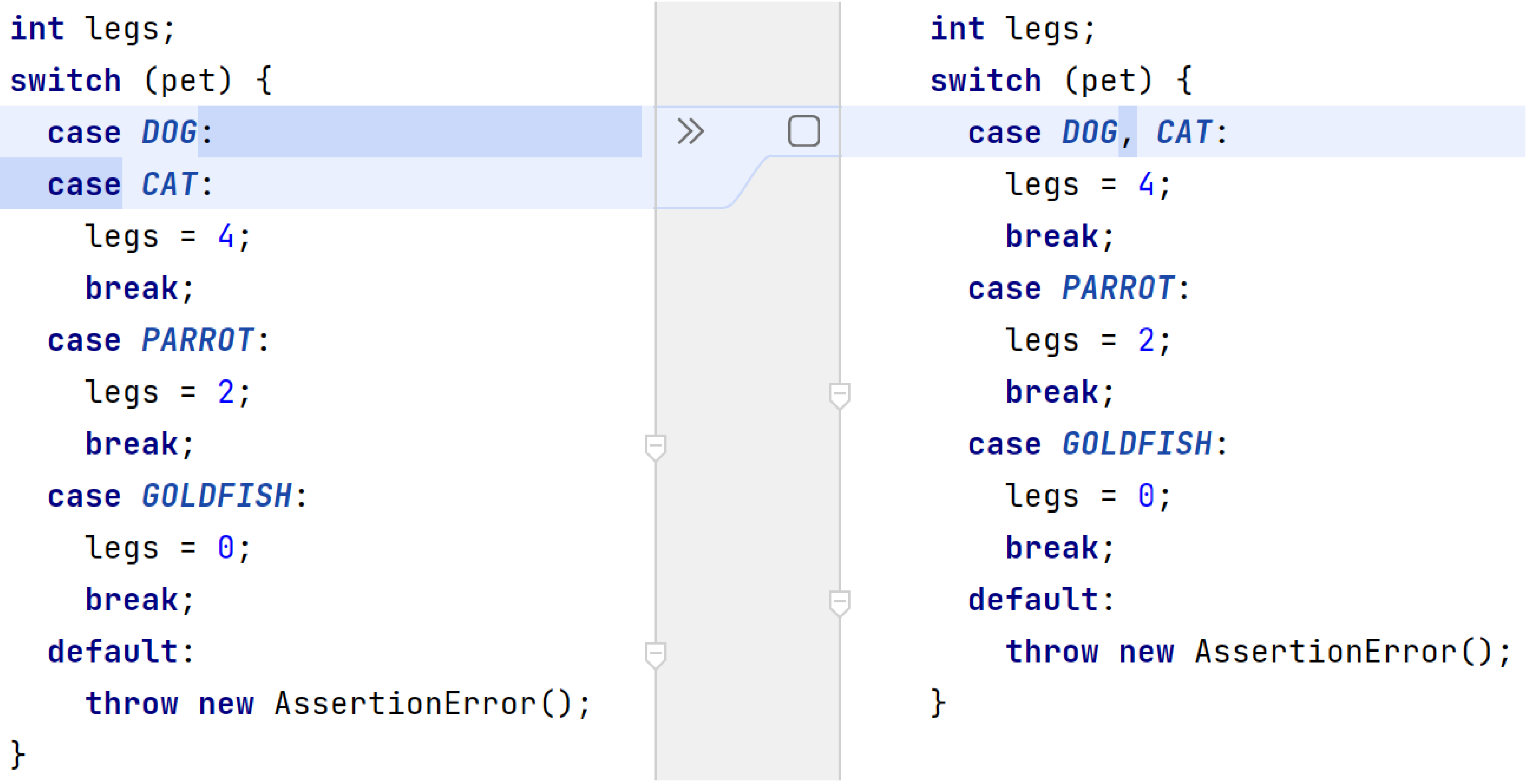
Switch expressions

Java 12: Preview

Java 14: Standard

```
enum Pet {
  DOG, CAT, PARROT, GOLDFISH
}

int legs;
switch (pet) {
  case DOG:
  case CAT:
    legs = 4;
    break;
  case PARROT:
    legs = 2;
    break;
  case GOLDFISH:
    legs = 0;
    break;
  default:
    throw new AssertionError();
}
```

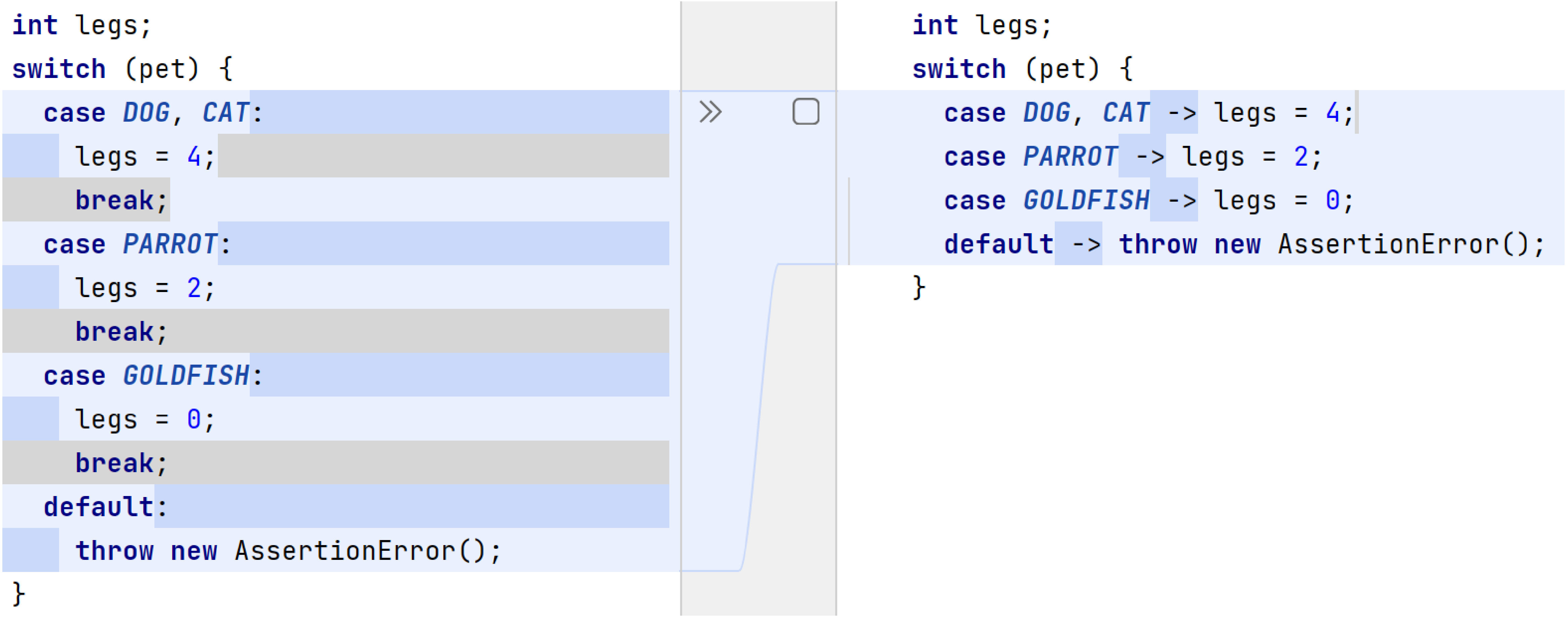
Switch expressions



```
int legs;
switch (pet) {
  case DOG:
  case CAT:
    legs = 4;
    break;
  case PARROT:
    legs = 2;
    break;
  case GOLDFISH:
    legs = 0;
    break;
  default:
    throw new AssertionError();
}
```

```
int legs;
switch (pet) {
  case DOG, CAT:
    legs = 4;
    break;
  case PARROT:
    legs = 2;
    break;
  case GOLDFISH:
    legs = 0;
    break;
  default:
    throw new AssertionError();
}
```

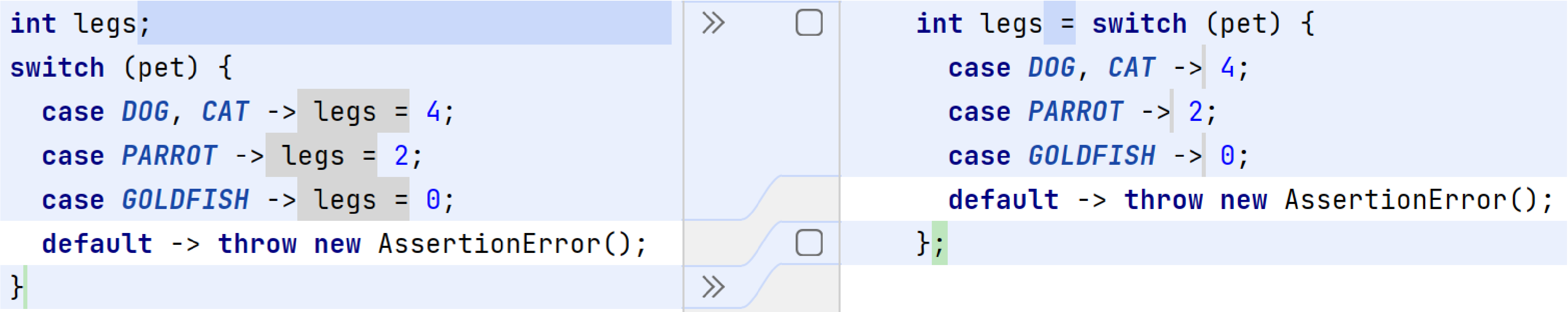
Switch expressions



```
int legs;
switch (pet) {
  case DOG, CAT:
    legs = 4;
    break;
  case PARROT:
    legs = 2;
    break;
  case GOLDFISH:
    legs = 0;
    break;
  default:
    throw new AssertionError();
}
```

```
int legs;
switch (pet) {
  case DOG, CAT -> legs = 4;
  case PARROT -> legs = 2;
  case GOLDFISH -> legs = 0;
  default -> throw new AssertionError();
}
```

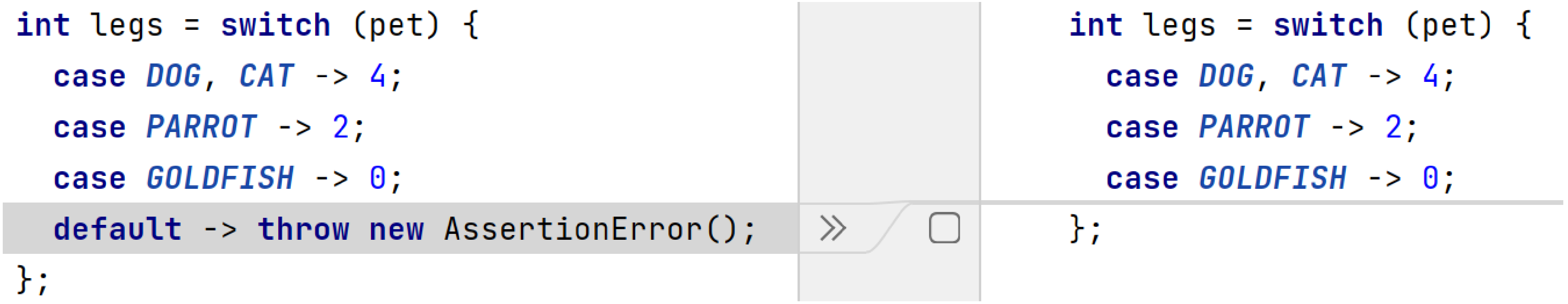
Switch expressions



```java
int legs;
switch (pet) {
  case DOG, CAT -> legs = 4;
  case PARROT -> legs = 2;
  case GOLDFISH -> legs = 0;
  default -> throw new AssertionError();
}
```

```java
int legs = switch (pet) {
  case DOG, CAT -> 4;
  case PARROT -> 2;
  case GOLDFISH -> 0;
  default -> throw new AssertionError();
};
```

Switch expressions

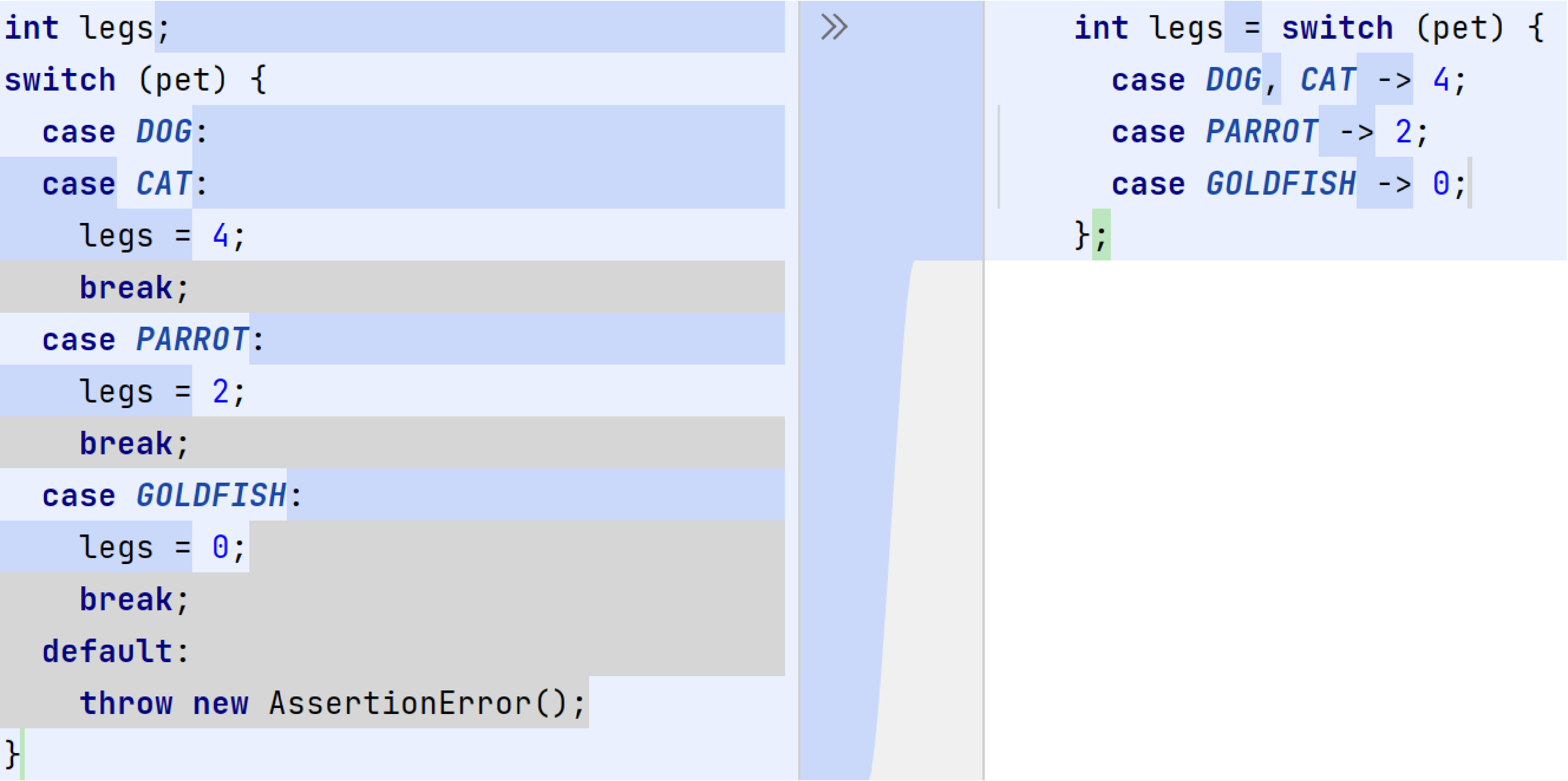


```java
int legs = switch (pet) {
  case DOG, CAT -> 4;
  case PARROT -> 2;
  case GOLDFISH -> 0;
  default -> throw new AssertionError();
};
```

```java
int legs = switch (pet) {
  case DOG, CAT -> 4;
  case PARROT -> 2;
  case GOLDFISH -> 0;
};
```

```java
enum Pet {
  DOG, CAT, PARROT, GOLDFISH
}
```

Switch expressions



```java
int legs;
switch (pet) {
  case DOG:
  case CAT:
    legs = 4;
    break;
  case PARROT:
    legs = 2;
    break;
  case GOLDFISH:
    legs = 0;
    break;
  default:
    throw new AssertionError();
}
```

```java
int legs = switch (pet) {
  case DOG, CAT -> 4;
  case PARROT -> 2;
  case GOLDFISH -> 0;
};
```

Switch expressions

```
int legs = switch (pet) {
  case DOG, CAT -> 4;
  case PARROT -> {
    System.out.println("Попка-дурак!");
    yield 2;
  }
  case GOLDFISH ->
    throw new IllegalArgumentException("Ноги у рыбов?! Красивое...");
};
```

Switch expressions

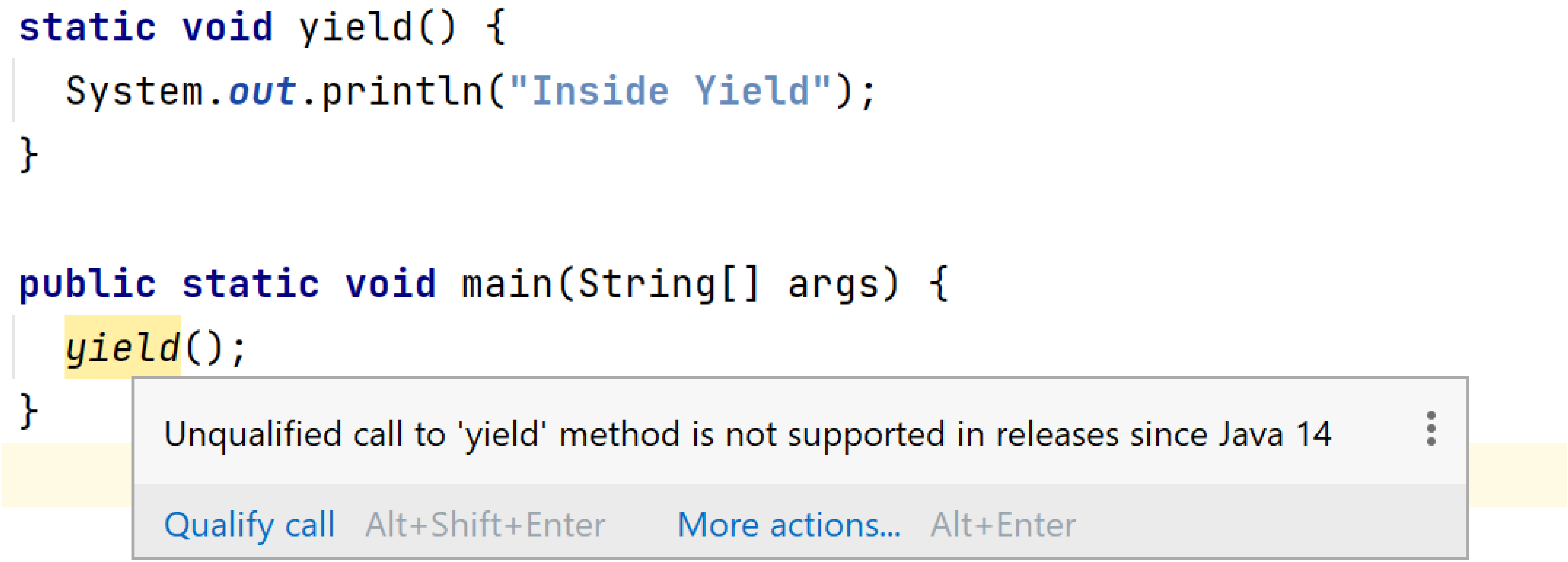
```
    public class Demo {
      static void yield() {
        System.out.println("Inside Yield");
      }

      public static void main(String[] args) {
❌        yield();
      }
    }
```

Switch expressions

✓

```
public class Demo {
  static void yield() {
    System.out.println("Inside Yield");
  }

  public static void main(String[] args) {
    Demo.yield();
  }
}
```

Java | Java language level issues | Forward compatibility



```
static void yield() {
  System.out.println("Inside Yield");
}

public static void main(String[] args) {
  yield();
}
```

Unqualified call to 'yield' method is not supported in releases since Java 14

Pattern matching for instanceof

Java 14: Preview
Java 16: Standard

```
public void processValue(Object obj) {
  if (obj instanceof String s) {
    System.out.println("String: " + s.trim());
  } else if (obj instanceof LocalDate date) {
    System.out.println("Date: " +
                       DateTimeFormatter.ISO_DATE.format(date));
  } else if (obj instanceof Number number) {
    System.out.println("Number: " + number.longValue());
  } else {
    System.out.println("Something else");
  }
}
```

Pattern matching for instanceof

```
if (obj instanceof String) {
  String s = (String) obj;
  System.out.println("String: " + s.trim());
} else if (obj instanceof LocalDate) {
  LocalDate date = (LocalDate) obj;
  System.out.println("Date: " +
                     DateTimeFormatter.ISO_DATE.format(date));
} else if (obj instanceof Number) {
  Number number = (Number) obj;
  System.out.println("Number: " + number.longValue());
} else {
  System.out.println("Something else");
}
```

Pattern matching for instanceof

```java
if (obj instanceof String s && !s.isEmpty()) {
  System.out.println("String: " + s.trim());
}
```

✓

Pattern matching for instanceof

```
if (obj instanceof String s && !s.isEmpty()) {
  System.out.println("String: " + s.trim());
}
```

```
if (!(obj instanceof String s) || s.isEmpty()) {
  return;
}
System.out.println("String: " + s.trim());
```

✓

Pattern matching for instanceof

```
if (obj instanceof String s && !s.isEmpty()) {
  System.out.println("String: " + s.trim());
}
```

```
if (!(obj instanceof String s) || s.isEmpty()) {
  return;
}
System.out.println("String: " + s.trim());
```

✓

Pattern matching for instanceof

```
if (!(obj instanceof String s) || s.isEmpty()) {
  return;
}
System.out.println("String: " + s.trim());
```

✓

```
if (!(obj instanceof String s) || s.isEmpty()) {
  System.out.println("Wrong object :(");
}
System.out.println("String: " + s.trim());
```

✗

Pattern matching for instanceof

```java
@Target(ElementType.LOCAL_VARIABLE)
@interface LocalAnno {}

void printIfString(Object obj) {
  if (obj instanceof @LocalAnno final String s) {
    System.out.println(s.trim());
  }
}
```

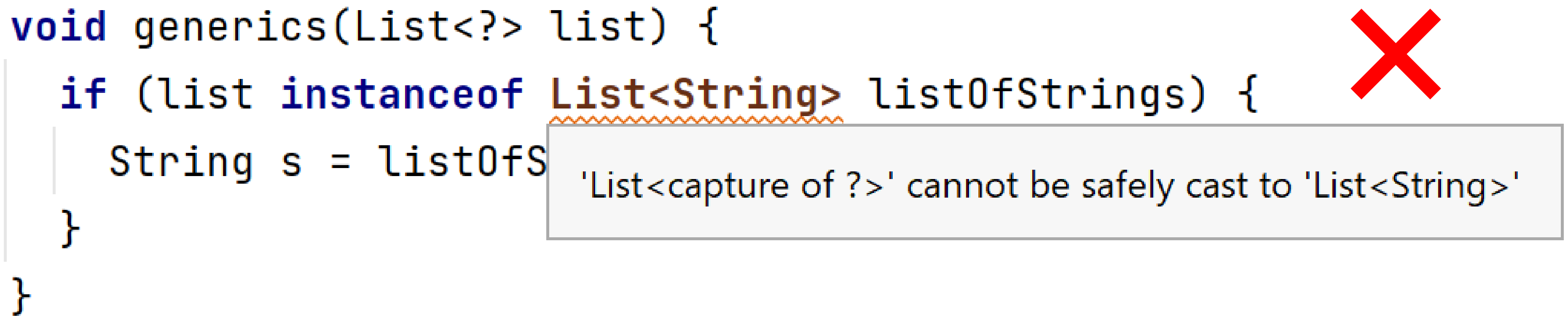
Pattern matching for instanceof

```
void generics(List<String> list) {
  if (list instanceof ArrayList<String> arrayList) {
    arrayList.trimToSize();
  }
}
```

✓

Pattern matching for instanceof

```
void generics(List<String> list) {
  if (list instanceof ArrayList<String> arrayList) {
    arrayList.trimToSize();
  }
}
```



```
void generics(List<?> list) {
  if (list instanceof List<String> listOfStrings) {
    String s = listOfS
  }
}
```

'List<capture of ?>' cannot be safely cast to 'List<String>'

Records

Java 14: Preview
Java 16: Standard

```
public record Point(int x, int y) {}
```

```java
public record Point(int x, int y) {}
```

```java
import java.util.Objects;

public final class Point {
  private final int x;
  private final int y;

  public Point(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int x() {
    return x;
  }

  public int y() {
    return y;
  }
```

```java
  @Override
  public boolean equals(Object obj) {
    if (obj == this) return true;
    if (obj == null || obj.getClass()
        != this.getClass()) return false;
    var that = (Point) obj;
    return this.x == that.x &&
           this.y == that.y;
  }

  @Override
  public int hashCode() {
    return Objects.hash(x, y);
  }

  @Override
  public String toString() {
    return "Point[" +
           "x=" + x + ", " +
           "y=" + y + ']';
  }
}
```

```java
public record Point(int x, int y) {}
```

```java
import java.util.Objects;

public final class Point {
  private final int x;
  private final int y;

  public Point(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int x() {
    return x;
  }

  public int y() {
    return y;
  }

  @Override
  public boolean equals(Object obj) {
    if (obj == this) return true;
    if (obj == null || obj.getClass()
        != this.getClass()) return false;
    var that = (Point) obj;
    return this.x == that.x &&
           this.y == that.y;
  }

  @Override
  public int hashCode() {
    return Objects.hash(x, y);
  }

  @Override
  public String toString() {
    return "Point[" +
           "x=" + x + ", " +
           "y=" + y + ']';
  }
}
```

```java
public record Point(int x, int y) {}
```

```java
import java.util.Objects;

public final class Point {
  private final int x;
  private final int y;

  public Point(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int x() {
    return x;
  }

  public int y() {
    return y;
  }

  @Override
  public boolean equals(Object obj) {
    if (obj == this) return true;
    if (obj == null || obj.getClass()
        != this.getClass()) return false;
    var that = (Point) obj;
    return this.x == that.x &&
           this.y == that.y;
  }

  @Override
  public int hashCode() {
    return Objects.hash(x, y);
  }

  @Override
  public String toString() {
    return "Point[" +
           "x=" + x + ", " +
           "y=" + y + ']';
  }
}
```

```
public record Point(int x, int y) {}
```

```
import java.util.Objects;

public final class Point {
  private final int x;
  private final int y;

  public Point(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int x() {
    return x;
  }

  public int y() {
    return y;
  }

  @Override
  public boolean equals(Object obj) {
    if (obj == this) return true;
    if (obj == null || obj.getClass()
        != this.getClass()) return false;
    var that = (Point) obj;
    return this.x == that.x &&
           this.y == that.y;
  }

  @Override
  public int hashCode() {
    return Objects.hash(x, y);
  }

  @Override
  public String toString() {
    return "Point[" +
           "x=" + x + ", " +
           "y=" + y + ']';
  }
}
```

Records

```
public record Point(int x, int y) {
  public Point(int x, int y) {
    if (x < 0 || y < 0) {
      throw new IllegalArgumentException();
    }
    this.x = x;
    this.y = y;
  }
}
```

Records

```
public record Point(int x, int y) {
  public Point(int x, int y) {
    if (x < 0 || y < 0) {
      throw new IllegalArgumentException();
    }
    this.x = x;
    this.y = y;
  }
}
```

```
public record Point(int x, int y) {
  public Point {
    if (x < 0 || y < 0) {
      throw new IllegalArgumentException();
    }
  }
}
```

Records

- ✓ Иммутабельны
- ✓ Всегда имеют канонический конструктор с параметрами, соответствующими компонентам
- ✓ Всегда имеет аксессоры с именами, соответствующими компонентам
- ✓ Два рекорда, сконструированные с одинаковыми параметрами, равны по equals и имеют равный hashCode
- ✓ Если считать компоненты через аксессоры и создать из них новый рекорд, он будет равен исходному по equals и иметь такой же hashCode

Records – reflection

```java
public record Point(int x, int y) {
  public static void main(String[] args) {
    boolean record = Point.class.isRecord();
    if (record) {
      RecordComponent[] components = Point.class.getRecordComponents();
      System.out.println(Arrays.toString(components));
      Method accessor = components[0].getAccessor();
      System.out.println(accessor);
    }
  }
}
```

```
[int x, int y]
public int Point.x()
```

Records – reflection

```java
public record Point(int x, int y) {
  public static void main(String[] args) {
    boolean record = Point.class.isRecord();
    if (record) {
      RecordComponent[] components = Point.class.getRecordComponents();
      System.out.println(Arrays.toString(components));
      Method accessor = components[0].getAccessor();
      System.out.println(accessor);
    }
  }
}
```

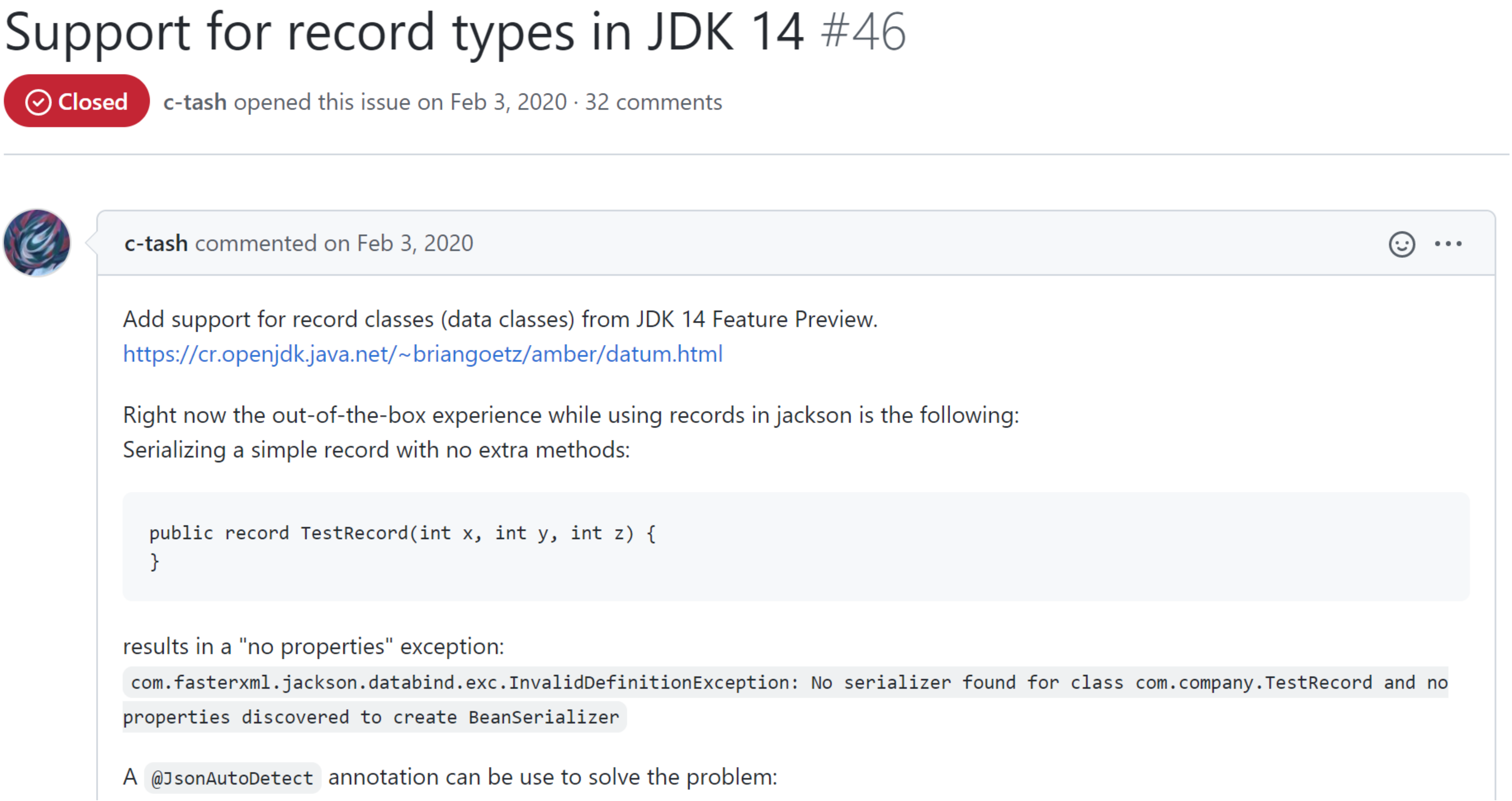
```
[int x, int y]
public int Point.x()
```

get! Смотрите, get!

Records – reflection

```
static <T extends Record> Constructor<T> canonicalConstructor(Class<T> cls)
  throws NoSuchMethodException {
  Class<?>[] paramTypes = Arrays.stream(cls.getRecordComponents())
    .map(RecordComponent::getType)
    .toArray(Class<?>[]::new);
  return cls.getDeclaredConstructor(paramTypes);
}
```

Я щас умру, а-ха-ха



Support for record types in JDK 14 #46

Closed c-tash opened this issue on Feb 3, 2020 · 32 comments

c-tash commented on Feb 3, 2020

Add support for record classes (data classes) from JDK 14 Feature Preview.
https://cr.openjdk.java.net/~briangoetz/amber/datum.html

Right now the out-of-the-box experience while using records in jackson is the following:
Serializing a simple record with no extra methods:

```
public record TestRecord(int x, int y, int z) {
}
```

results in a "no properties" exception:
`com.fasterxml.jackson.databind.exc.InvalidDefinitionException: No serializer found for class com.company.TestRecord and no properties discovered to create BeanSerializer`

A `@JsonAutoDetect` annotation can be use to solve the problem:

https://github.com/FasterXML/jackson-future-ideas/issues/46

```java
public record Point(int x, int y) implements Serializable {
  public static void main(String[] args) throws IOException {
    Point point = new Point(-1, -1);
    var result = new ByteArrayOutputStream();
    try (var oos = new ObjectOutputStream(result)) {
      oos.writeObject(point);
    }
    System.out.println(Base64.getEncoder().encodeToString(result.toByteArray()));
  }
}
```

rO0ABXNyAAVQb2ludAAAAAAAAAAAAgACSQABeEkAAXl4cP//////////

```java
public record Point(int x, int y) implements Serializable {
  public static void main(String[] args) throws IOException {
    Point point = new Point(-1, -1);
    var result = new ByteArrayOutputStream();
    try (var oos = new ObjectOutputStream(result)) {
      oos.writeObject(point);
    }
    System.out.println(Base64.getEncoder().encodeToString(result.toByteArray()));
  }
}
```

```
rO0ABXNyAAVQb2ludAAAAAAAAAAAAgACSQABeEkAAXl4cP//////////
```

```java
public record Point(int x, int y) implements Serializable {
  public static void main(String[] args) throws IOException, ClassNotFoundException {
    var input = Base64.getDecoder()
      .decode("rO0ABXNyAAVQb2ludAAAAAAAAAAAAgACSQABeEkAAXl4cP//////////");
    try (var ois = new ObjectInputStream(new ByteArrayInputStream(input))) {
      Point point = (Point) ois.readObject();
      System.out.println(point);
    }
  }
}
```

```
Point[x=-1, y=-1]
```

```java
public record Point(int x, int y) implements Serializable {
  public Point {
    if (x < 0 || y < 0) {
      throw new IllegalArgumentException();
    }
  }

  public static void main(String[] args) throws IOException, ClassNotFoundException {
    var input = Base64.getDecoder()
      .decode("rO0ABXNyAAVQb2ludAAAAAAAAAAAAgACSQABeEkAAXl4cP//////////");
    try (var ois = new ObjectInputStream(new ByteArrayInputStream(input))) {
      Point point = (Point) ois.readObject();
      System.out.println(point);
    }
  }
}
```

```
Exception in thread "main" java.io.InvalidObjectException
	at java.io.ObjectInputStream.readRecord(ObjectInputStream.java:2348)
	at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2236)
	at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1742)
	at java.io.ObjectInputStream.readObject(ObjectInputStream.java:514)
	at java.io.ObjectInputStream.readObject(ObjectInputStream.java:472)
	at Point.main(Point.java:20)
Caused by: java.lang.IllegalArgumentException
	at Point.<init>(Point.java:12)
	at java.io.ObjectInputStream.readRecord(ObjectInputStream.java:2346)
	... 5 more
```

Records

```
List<Customer> findTopScoredCustomers(List<Customer> allCustomers) {
  return allCustomers.stream()
    .sorted(comparing(customer -> calculateScore(customer), reverseOrder()))
    .limit(10)
    .toList();
}
```

Records

```
List<Customer> findTopScoredCustomers(List<Customer> allCustomers) {
  return allCustomers.stream()
    .sorted(comparing(customer -> calculateScore(customer), reverseOrder()))
    .limit(10)
    .toList(); // ~~.collect(Collectors.toList())~~
}
```

Records

```
List<Customer> findTopScoredCustomers(List<Customer> allCustomers) {
  return allCustomers.stream()
    .sorted(comparing(customer -> calculateScore(customer), reverseOrder()))
    .limit(10)
    .toList();
}
```

```
List<Customer> findTopScoredCustomers(List<Customer> allCustomers) {
  record CustomerAndScore(Customer customer, double score) {}

  return allCustomers.stream()
    .map(c -> new CustomerAndScore(c, calculateScore(c)))
    .sorted(comparing(CustomerAndScore::score, reverseOrder()))
    .map(CustomerAndScore::customer)
    .limit(10)
    .toList();
}
```

Records

```
Pair<Integer, Integer> pair;
```

✗

```
record Interval(int start, int end) {}
record Point(int x, int y) {}
record Fraction(int numerator, int denominator) {}
```

✓

Records

```java
record Pair<T1, T2>(T1 t1, T2 t2) {}
Pair<Integer, Integer> pair;
```

✗

```java
record Interval(int start, int end) {}
record Point(int x, int y) {}
record Fraction(int numerator, int denominator) {}
```

✓

Records

```
public static void main(String[] args) {
  record Vector(double x, double y, double z) {
    void test() {
      System.out.println(Arrays.toString(args));
    }
  }
}
```

Records

```
public static void main(String[] args) {
  record Vector(double x, double y, double z) {
    void test() {
      System.out.println(Arrays.toString(args)); ✗
    }
  }
}
```

Local interfaces, enums, inner statics

```
void test() {
  enum LocalEnum {A, B, C}
  interface LocalInterface {
    void test(LocalEnum e);
  }
  LocalInterface r = new LocalInterface() {
    static final AtomicInteger callCounter = new AtomicInteger();

    @Override
    public void test(LocalEnum e) {
      System.out.println(callCounter.incrementAndGet());
      System.out.println(e);
      staticInAnonymous();
    }

    static void staticInAnonymous() {
      System.out.println("I'm static");
    }
  };
  r.test(LocalEnum.A);
}
```

Sealed classes

Java 15: Preview
Java 17: Standard

```
public abstract sealed class Pet permits Cat, Dog, Parrot, Goldfish {
  …
}
```

Sealed classes

```
public abstract sealed class Pet permits Cat, Dog, Parrot, Goldfish {
  …
}

public sealed class Dog extends Pet {
  public static final class Dachshund extends Dog {}
  public static final class ChowChow extends Dog {}
  public static final class Collie extends Dog {}
}

public final class Goldfish extends Pet { }

public final class Parrot extends Pet { }

public non-sealed class Cat extends Pet { }
```

Sealed interfaces

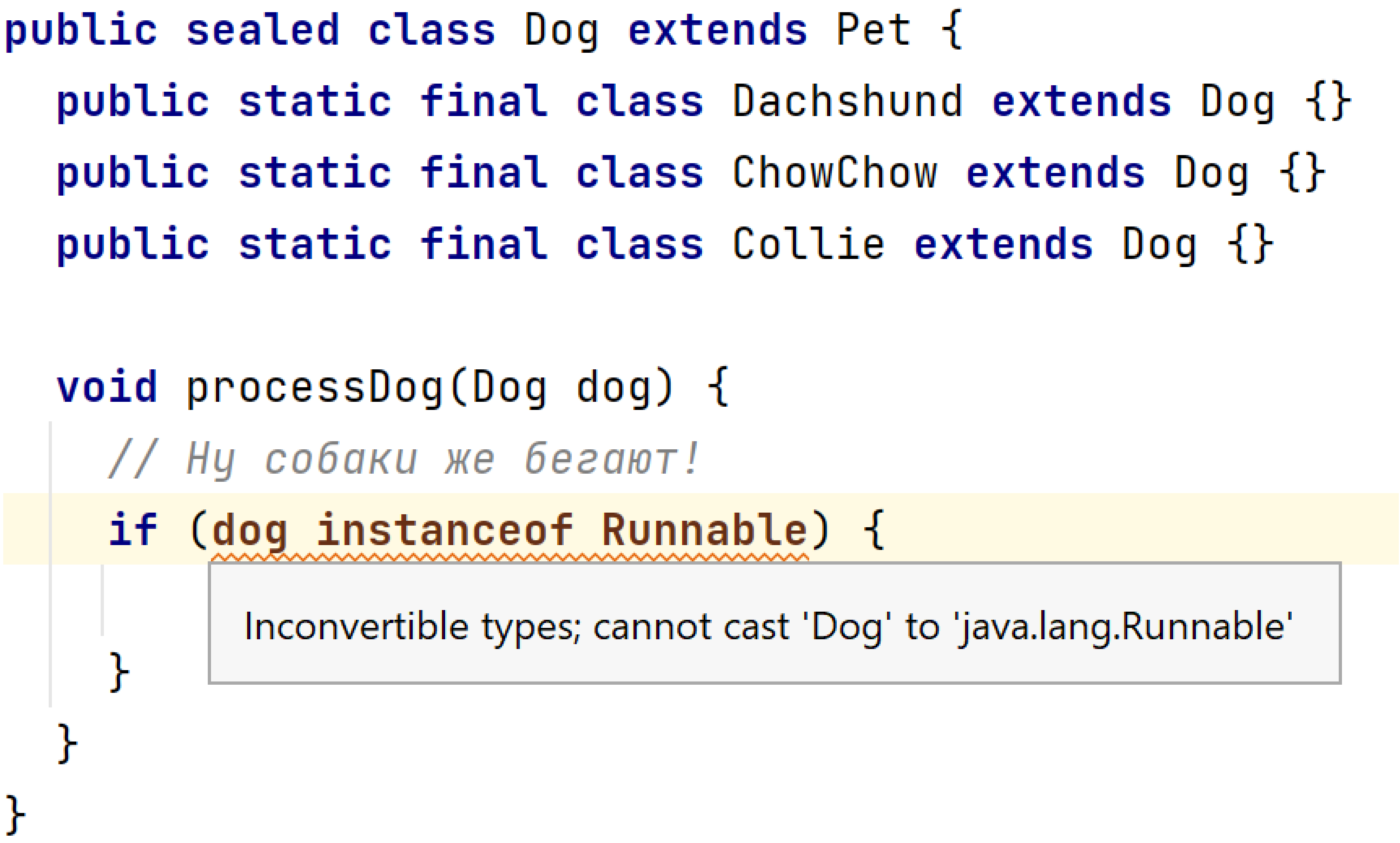
```
public sealed interface Pet permits Cat, Dog, Parrot, Goldfish {
  …
}
```

Sealed classes – reflection

```java
public static void main(String[] args) {
  boolean sealed = Pet.class.isSealed();
  if (sealed) {
    Class<?>[] subclasses = Pet.class.getPermittedSubclasses();
    System.out.println(Arrays.toString(subclasses));
  }
}
```

```
[class Cat, class Dog, class Parrot, class Goldfish]
```

Sealed classes



```
public sealed class Dog extends Pet {
  public static final class Dachshund extends Dog {}
  public static final class ChowChow extends Dog {}
  public static final class Collie extends Dog {}

  void processDog(Dog dog) {
    // Ну собаки же бегают!
    if (dog instanceof Runnable) {

    }
  }
}
```

Inconvertible types; cannot cast 'Dog' to 'java.lang.Runnable'

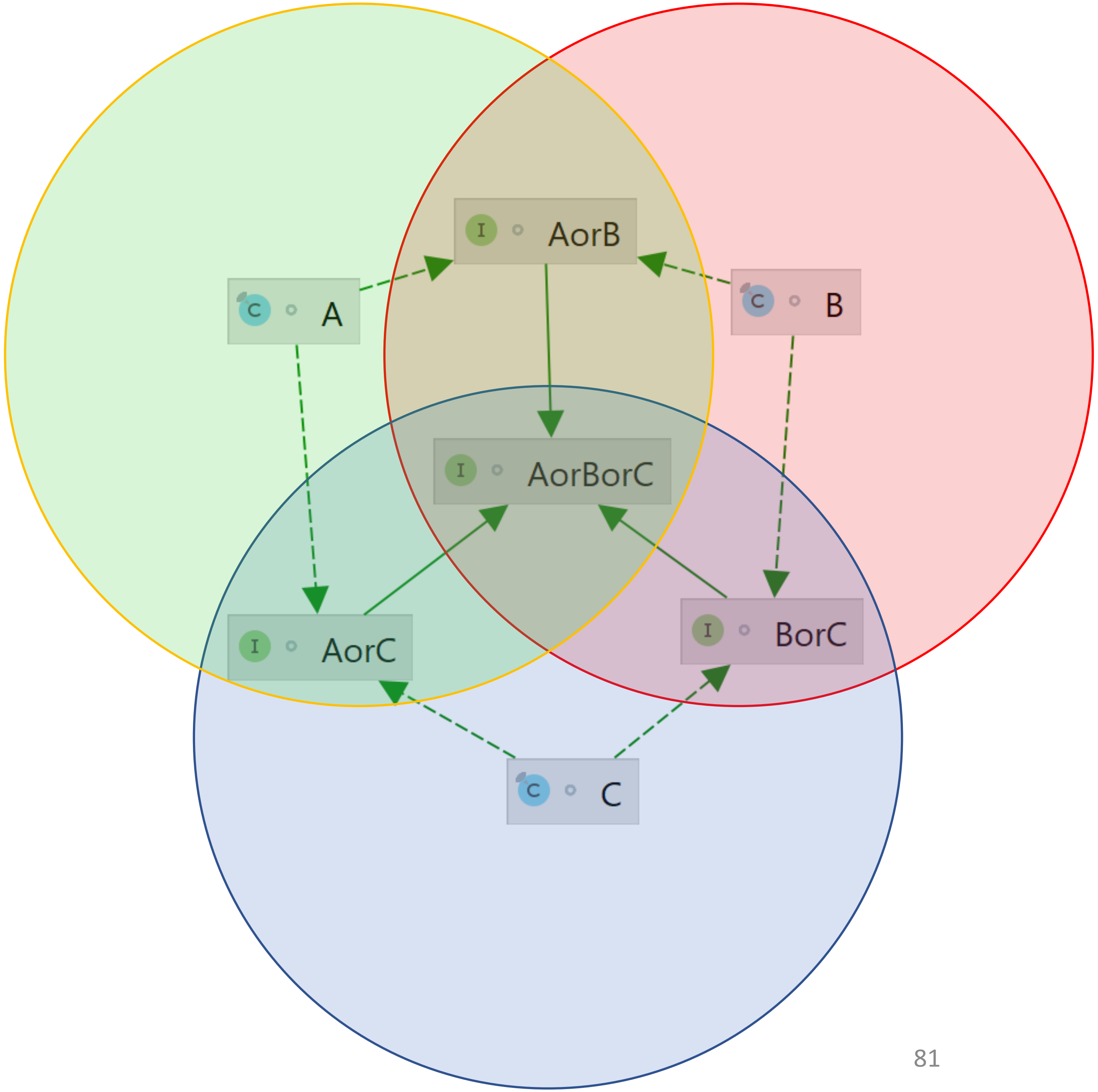
Algebraic data types*

```
// Тип-сумма
sealed interface XOrY {}
// Тип-произведение
record XAndY(X x, Y y) {}

final class X implements XOrY {}
final class Y implements XOrY {}
```

* Как бы.

Algebraic data types*

```
sealed interface AorBorC {}
sealed interface AorB extends AorBorC {}
sealed interface BorC extends AorBorC {}
sealed interface AorC extends AorBorC {}
final class A implements AorB, AorC {}
final class B implements AorB, BorC {}
final class C implements AorC, BorC {}
```



* Как бы.

Pattern matching for switch

Java 17: Preview

```
static String asString(Object value) {
  return switch (value) {
    case Enum<?> e -> e.getDeclaringClass().getSimpleName() + "." + e.name();
    case Collection c -> "Collection [size = %d]".formatted(c.size());
    case Object[] arr -> "Array [length = %d]".formatted(arr.length);
    case String s && s.length() > 50 -> '"'+s.substring(0, 50)+"...\"";
    case String s -> '"' + s + '"';
    case null -> "null";
    default -> value.toString();
  };
}
```

Pattern matching for switch

Java 17: Preview

```java
static String asString(Object value) {
  return switch (value) {
    case Enum<?> e -> e.getDeclaringClass().getSimpleName() + "." + e.name();
    case Collection c -> "Collection [size = %d]".formatted(c.size());
    case Object[] arr -> "Array [length = %d]".formatted(arr.length);
    case String s && s.length() > 50 -> '"'+s.substring(0, 50)+"...\"";
    case String s -> '"' + s + '"';
    case null -> "null";
    default -> value.toString();
  };
}
```

Pattern matching for switch

Java 17: Preview

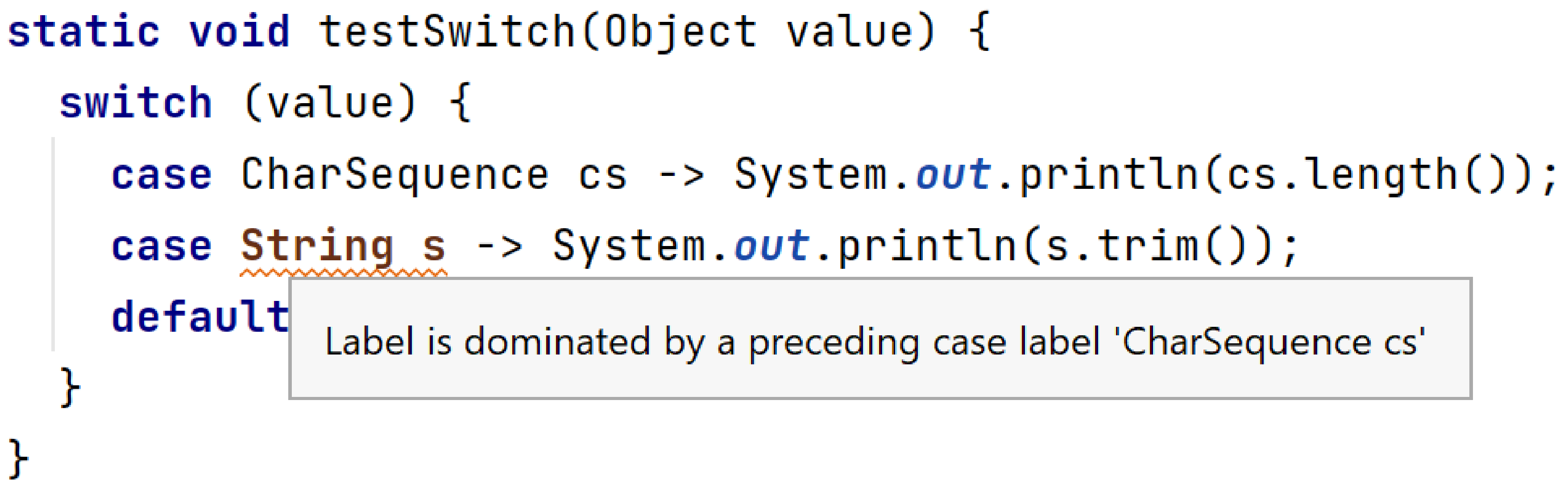
```
static String asString(Object value) {
  return switch (value) {
    case Enum<?> e -> e.getDeclaringClass().getSimpleName() + "." + e.name();
    case Collection c -> "Collection [size = %d]".formatted(c.size());
    case Object[] arr -> "Array [length = %d]".formatted(arr.length);
    case String s && s.length() > 50 -> '"'+s.substring(0, 50)+"...\"";
    case String s -> '"' + s + '"';
    case null -> "null";
    default -> value.toString();
  };
}
```

Pattern matching for switch

Java 17: Preview

```java
static String asString(Object value) {
  return switch (value) {
    case Enum<?> e -> e.getDeclaringClass().getSimpleName() + "." + e.name();
    case Collection c -> "Collection [size = %d]".formatted(c.size());
    case Object[] arr -> "Array [length = %d]".formatted(arr.length);
    case String s && s.length() > 50 -> '"'+s.substring(0, 50)+"...\"";
    case String s -> '"' + s + '"';
    case null -> "null";
    default -> value.toString();
  };
}
```

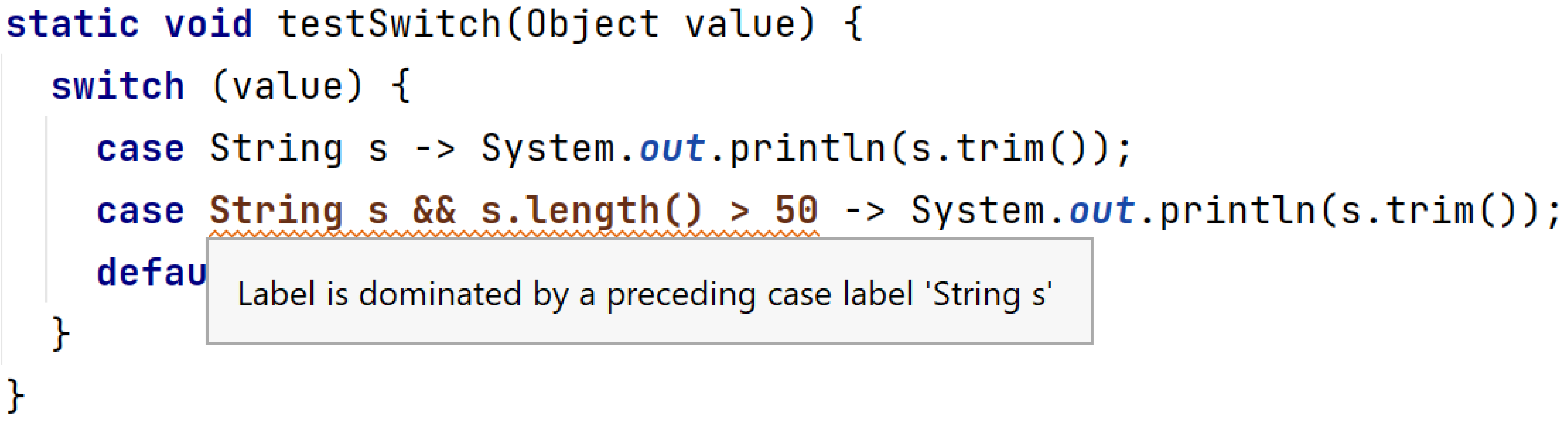
Pattern matching for switch



```
static void testSwitch(Object value) {
  switch (value) {
    case CharSequence cs -> System.out.println(cs.length());
    case String s -> System.out.println(s.trim());
    default
  }
}
```

Label is dominated by a preceding case label 'CharSequence cs'

Pattern matching for switch



```
static void testSwitch(Object value) {
  switch (value) {
    case String s -> System.out.println(s.trim());
    case String s && s.length() > 50 -> System.out.println(s.trim());
    defau
  }
}
```

Label is dominated by a preceding case label 'String s'

Pattern matching for switch

```
static void testSwitch(Object value) {
  switch (value) {
    case null, String s -> System.out.println(s);
    default -> {}
  }
}
```

Pattern matching for switch

```java
void switchOverStrings(String s) {
  switch (s) {
    case "One" -> System.out.println(1);
    case "Two" -> System.out.println(2);
  }
}
```

✓

Pattern matching for switch

```
void switchOverStrings(String s) {
  switch (s) {
    case "One" -> System.out.println(1);
    case "Two" -> System.out.println(2);
  }
}
```

✓

```
void switchOverStrings(String s) {
  switch (s) {
    case "One" -> System.out.println(1);
    case "Two" -> System.out.println(2);
    case null -> System.out.println("null");
  }
}
```

✗

Pattern matching for switch

```java
void switchOverStrings(String s) {
  switch (s) {
    case "One" -> System.out.println(1);
    case "Two" -> System.out.println(2);
  }
}
```

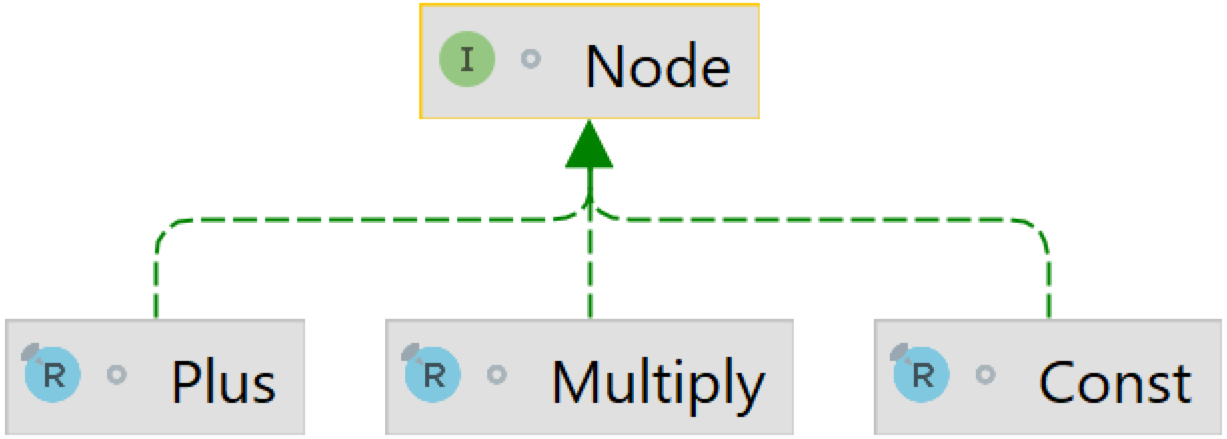
✓

```java
void switchOverStrings(String s) {
  switch (s) {
    case "One" -> System.out.println(1);
    case "Two" -> System.out.println(2);
    case null -> System.out.println("null");
    default -> throw new IllegalStateException("Unexpected value: " + s);
  }
}
```

✓

Pattern matching for switch

```
sealed interface Node {}
record Const(int value) implements Node {}
record Plus(Node left, Node right) implements Node {}
record Multiply(Node left, Node right) implements Node {}
```



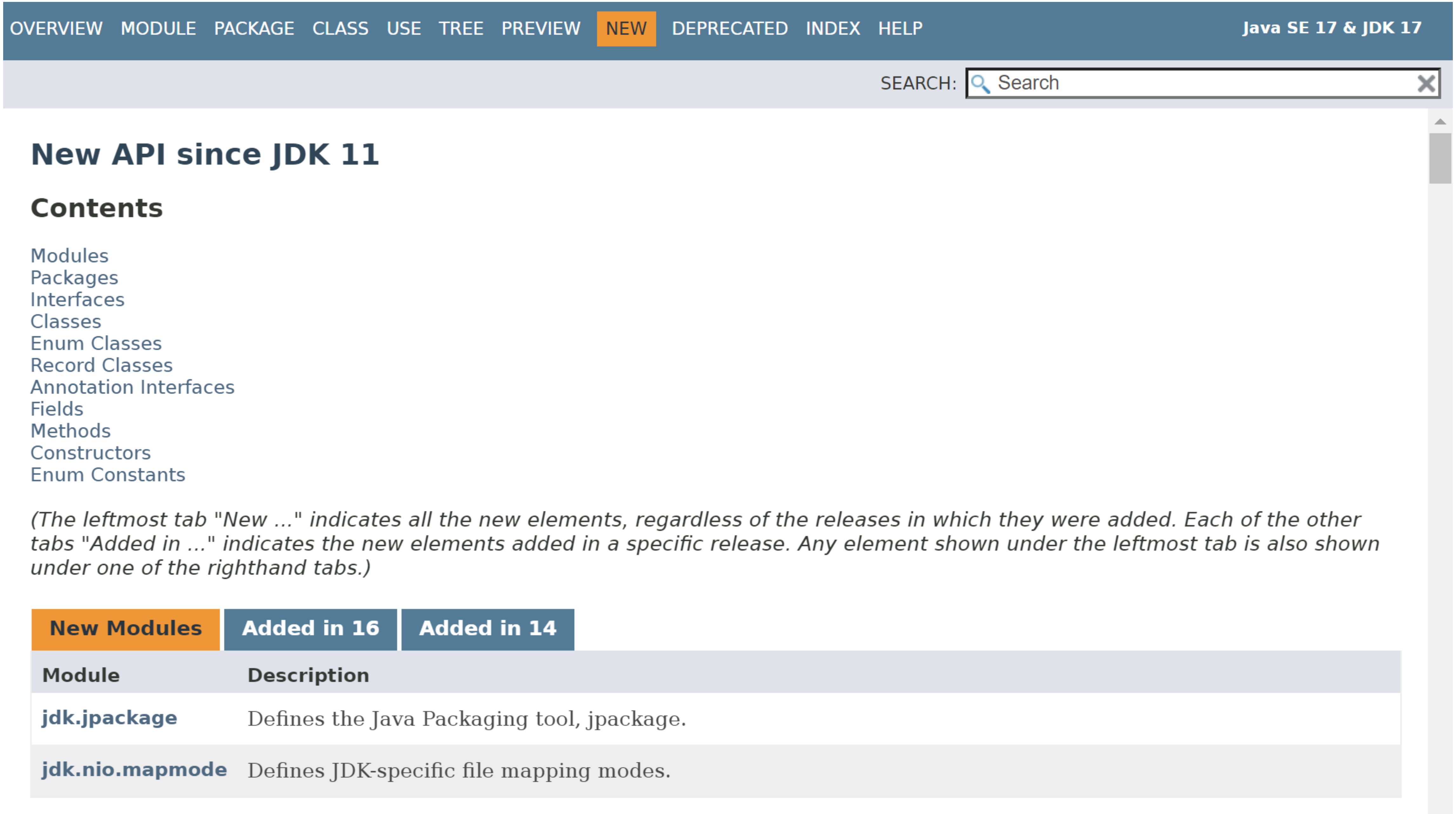
```java
interface NodeVisitor<T> {
  T visitConst(Const c);
  T visitPlus(Plus p);
  T visitMultiply(Multiply m);
}
sealed interface Node {
  <T> T accept(NodeVisitor<T> visitor);
}
record Const(int value) implements Node {
  public <T> T accept(NodeVisitor<T> visitor) {return visitor.visitConst(this);}
}
record Plus(Node left, Node right) implements Node {
  public <T> T accept(NodeVisitor<T> visitor) {return visitor.visitPlus(this);}
}
record Multiply(Node left, Node right) implements Node {
  public <T> T accept(NodeVisitor<T> visitor) {return visitor.visitMultiply(this);}
}
```

```java
static int evaluate(Node node) {
  return node.accept(new NodeVisitor<>() {
    public Integer visitConst(Const c) {
      return c.value();
    }

    public Integer visitPlus(Plus p) {
      return evaluate(p.left()) + evaluate(p.right());
    }

    public Integer visitMultiply(Multiply m) {
      return evaluate(m.left()) * evaluate(m.right());
    }
  });
}
```

```
static int evaluate(Node node) {
  if (node instanceof Const c) {
    return c.value();
  } else if (node instanceof Plus p) {
    return evaluate(p.left()) + evaluate(p.right());
  } else if (node instanceof Multiply m) {
    return evaluate(m.left()) * evaluate(m.right());
  }
  throw new AssertionError("Unknown node type: " + node.getClass());
}
```

```java
static int evaluate(Node node) {
  if (node instanceof Const c) {
    return c.value();
  } else if (node instanceof Plus p) {
    return evaluate(p.left()) + evaluate(p.right());
  } else if (node instanceof Multiply m) {
    return evaluate(m.left()) * evaluate(m.right());
  }
  throw new AssertionError("Unknown node type: " + node.getClass());
}
```

```java
static int evaluate(Node node) {
  return switch (node) {
    case Const c -> c.value();
    case Plus p -> evaluate(p.left()) + evaluate(p.right());
    case Multiply m -> evaluate(m.left()) * evaluate(m.right());
  };
}
```



New API since JDK 11

Contents

(The leftmost tab "New ..." indicates all the new elements, regardless of the releases in which they were added. Each of the other tabs "Added in ..." indicates the new elements added in a specific release. Any element shown under the leftmost tab is also shown under one of the righthand tabs.)

Module	Description
jdk.jpackage	Defines the Java Packaging tool, jpackage.
jdk.nio.mapmode	Defines JDK-specific file mapping modes.

String.indent (Java 12)

```
static String format(Node node) {
  return switch (node) {
    case Const c ->
      c.value() + "\n";
    case Plus p ->
      "+" + format(p.left()).indent(2).substring(1) + format(p.right()).indent(2);
    case Multiply m ->
      "*" + format(m.left()).indent(2).substring(1) + format(m.right()).indent(2);
  };
}

var expr = new Multiply(new Const(3), new Plus(new Const(4), new Const(5)));
System.out.println(format(expr));
```

```
* 3
  + 4
    5
```

String.stripIndent (Java 15)

```
String str = "  public class Hello {\n" +
             "    public static void main(String[] args) {\n" +
             "      System.out.println(\"Hello World!\");\n" +
             "    }\n" +
             "  }";
System.out.println(str.stripIndent());
```

```
public class Hello {
  public static void main(String[] args) {
    System.out.println("Hello World!");
  }
}
```

String.translateEscapes (Java 15)

```java
String str = "\\tHello\\n\\tWorld";
System.out.println(str);
System.out.println(str.translateEscapes());
```

```
\tHello\n\tWorld
      Hello
      World
```

Math.absExact (Java 15)

```
System.out.println(Math.absExact(Integer.MIN_VALUE));
```

```
Exception in thread "main" java.lang.ArithmeticException: Overflow to represent
absolute value of Integer.MIN_VALUE
       at java.base/java.lang.Math.absExact(Math.java:1448)
```

Stream.teeing (Java 12)

```
Collector<BigDecimal, ?, BigDecimal> averaging =
  Collectors.teeing(
    // collector1
    Collectors.reducing(BigDecimal.ZERO, BigDecimal::add),
    // collector2
    Collectors.counting(),
    // merger
    (sum, count) -> sum.divide(BigDecimal.valueOf(count),
                               2, RoundingMode.HALF_EVEN));

BigDecimal average = Stream.of(BigDecimal.ONE, BigDecimal.TEN)
                           .collect(averaging);
System.out.println(average); // 5.50
```

Stream.mapMulti (Java 16)

```
Stream.of("hello", "world")
    .<Character>mapMulti((str, sink) -> {
      for (char c : str.toCharArray()) {
        sink.accept(c);
      }
    })
    .forEach(System.out::println);
```

Stream.mapMulti (Java 16)

```
Stream.of("hello", "world")
    .<Character>mapMulti((str, sink) -> {
      for (char c : str.toCharArray()) {
        sink.accept(c);
      }
    })
    .forEach(System.out::println);
```

```
Stream.of("hello", "world")
    .flatMap(s -> s.chars().mapToObj(c -> (char) c))
    .forEach(System.out::println);
```

Stream.mapMulti (Java 16)

```java
List<Optional<String>> optionals =
  IntStream.range(0, 1000)
    .mapToObj(i -> Optional.of(i).filter(v -> v % 2 == 0).map(String::valueOf))
    .toList();
```

Stream.mapMulti (Java 16)

```java
List<Optional<String>> optionals =
  IntStream.range(0, 1000)
    .mapToObj(i -> Optional.of(i).filter(v -> v % 2 == 0).map(String::valueOf))
    .toList();

int totalLengthFilterMap() {
  return optionals.stream()
    .filter(Optional::isPresent)
    .map(Optional::get)
    .mapToInt(String::length)
    .sum();
}
```

Stream.mapMulti (Java 16)

```java
List<Optional<String>> optionals =
  IntStream.range(0, 1000)
    .mapToObj(i -> Optional.of(i).filter(v -> v % 2 == 0).map(String::valueOf))
    .toList();

int totalLengthFlatMap() {
  return optionals.stream()
    .flatMap(Optional::stream)
    .mapToInt(String::length)
    .sum();
}
```

Stream.mapMulti (Java 16)

```
List<Optional<String>> optionals =
  IntStream.range(0, 1000)
    .mapToObj(i -> Optional.of(i).filter(v -> v % 2 == 0).map(String::valueOf))
    .toList();

int totalLengthMapMulti() {
  return optionals.stream()
    .<String>mapMulti(Optional::ifPresent)
    .mapToInt(String::length)
    .sum();
}
```

БЕНЧМАРКАЕШЬ НЕБОСЬ?

```
Benchmark                Score   Error Units
totalLengthFilterMap     3.636 ± 0.060 μs/op
totalLengthFlatMap      14.233 ± 0.187 μs/op
totalLengthMapMulti      1.507 ± 0.016 μs/op
```

```
Benchmark                Score   Error Units
totalLengthFilterMap     3.636 ± 0.060 µs/op
totalLengthFlatMap      14.233 ± 0.187 µs/op
totalLengthMapMulti      1.507 ± 0.016 µs/op
```

```
Benchmark                                Score Units
totalLengthFilterMap:·gc.alloc.rate.norm   416 B/op
totalLengthFlatMap:·gc.alloc.rate.norm   68331 B/op
totalLengthMapMulti:·gc.alloc.rate.norm    328 B/op
```

```
Benchmark                  Score   Error Units
totalLengthFilterMap       3.636 ± 0.060 µs/op
totalLengthFlatMap        14.233 ± 0.187 µs/op
totalLengthMapMulti        1.507 ± 0.016 µs/op
totalLengthPlain           1.429 ± 0.025 µs/op
```

```
Benchmark                                      Score Units
totalLengthFilterMap:·gc.alloc.rate.norm         416 B/op
totalLengthFlatMap:·gc.alloc.rate.norm         68331 B/op
totalLengthMapMulti:·gc.alloc.rate.norm          328 B/op
totalLengthPlain:·gc.alloc.rate.norm               0 B/op
```

```java
int totalLengthPlain() {
  int sum = 0;
  for (var opt : optionals) {
    if (opt.isPresent()) {
      sum += opt.get().length();
    }
  }
  return sum;
}
```

HexFormat (Java 17)

```java
HexFormat format = HexFormat
  .ofDelimiter(":")
  .withUpperCase()
  .withPrefix("[")
  .withSuffix("]");
byte[] input = {(byte) 0xCA, (byte) 0xFE, (byte) 0xBA, (byte) 0xBE};
String asString = format.formatHex(input);
System.out.println(asString);
// [CA]:[FE]:[BA]:[BE]
byte[] output = format.parseHex(asString);
assert Arrays.equals(input, output);
```

ByteBuffer absolute positioning (Java 13, 16)

```
ByteBuffer.slice(index, length);
ByteBuffer.get(index, byteArray);
ByteBuffer.get(index, byteArray, offset, length);
ByteBuffer.put(index, byteArray);
ByteBuffer.put(index, byteArray, offset, length);
ByteBuffer.put(index, byteBuffer, offset, length);

CharBuffer/DoubleBuffer/FloatBuffer/IntBuffer/LongBuffer/ShortBuffer
```

Objects.check* для long (Java 16)

```
Objects.checkIndex(long index, long length);
Objects.checkFromToIndex(long fromIndex, long toIndex, long length);
Objects.checkFromIndexSize(long fromIndex, long size, long length);
```

Objects.check* для long (Java 16)

```
Objects.checkIndex(long index, long length);
Objects.checkFromToIndex(long fromIndex, long toIndex, long length);
Objects.checkFromIndexSize(long fromIndex, long size, long length);
```

RandomGeneratorFactory (Java 17)

```java
RandomGenerator generator = RandomGeneratorFactory
  .of("L128X1024MixRandom")
  .create();

int randomNumber = generator.nextInt(1, 100);

System.out.println("Extremely good random number between 1 and 100: "
                   + randomNumber);
```

RandomGeneratorFactory (Java 17)

Algorithm	Group	Period	StateBits	Equidistribution
L128X1024MixRandom	LXM	$2^{128}(2^{1024}-1)$	1152	1
L128X128MixRandom	LXM	$2^{128}(2^{128}-1)$	256	1
L128X256MixRandom	LXM	$2^{128}(2^{256}-1)$	384	1
L32X64MixRandom	LXM	$2^{32}(2^{64}-1)$	96	1
L64X1024MixRandom	LXM	$2^{64}(2^{1024}-1)$	1088	16
L64X128MixRandom	LXM	$2^{64}(2^{128}-1)$	192	2
L64X128StarStarRandom	LXM	$2^{64}(2^{128}-1)$	192	2
L64X256MixRandom	LXM	$2^{64}(2^{256}-1)$	320	4
Random	Legacy	2^{48}	48	0
SplittableRandom	Legacy	2^{64}	64	1
ThreadLocalRandom	Legacy	2^{64}	64	1
Xoroshiro128PlusPlus	Xoroshiro	$2^{128}-1$	128	1
Xoshiro256PlusPlus	Xoshiro	$2^{256}-1$	256	3

RandomGeneratorFactory (Java 17)

```
RandomGeneratorFactory
  .all()
  .map(g -> "%-25s|%10d|%10d|%.3g".formatted(g.name(), g.stateBits(),
                     g.equidistribution(), new BigDecimal(g.period())))
  .sorted()
  .forEach(System.out::println);
```

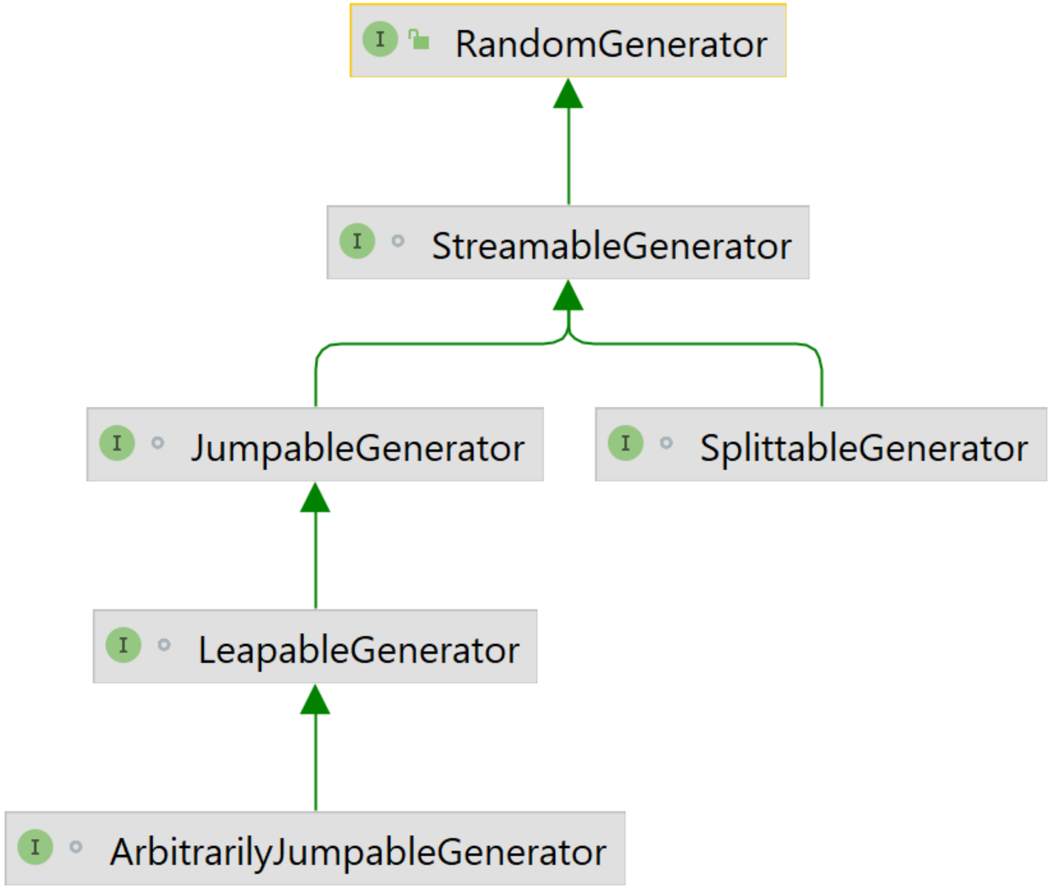
RandomGeneratorFactory (Java 17)

```
L128X1024MixRandom       |      1152|         1|6.12e+346
L128X128MixRandom        |       256|         1|1.16e+77
L128X256MixRandom        |       384|         1|3.94e+115
L32X64MixRandom          |        96|         1|7.92e+28
L64X1024MixRandom        |      1088|        16|3.32e+327
L64X128MixRandom         |       192|         2|6.28e+57
L64X128StarStarRandom    |       192|         2|6.28e+57
L64X256MixRandom         |       320|         4|2.14e+96
Random                   |        48|         0|2.81e+14
SecureRandom             |2147483647|2147483647|0.00
SplittableRandom         |        64|         1|1.84e+19
Xoroshiro128PlusPlus     |       128|         1|3.40e+38
Xoshiro256PlusPlus       |       256|         3|1.16e+77
```

interface RandomGenerator (Java 17)

```
DoubleStream doubles(…)
IntStream ints(…)
LongStream longs(…)
boolean nextBoolean()
void nextBytes(byte[])
float nextFloat(…)
double nextDouble(…)
int nextInt(…)
long nextLong(…)
double nextGaussian(…)
double nextExponential()
```

interface RandomGenerator (Java 17)



Reference.refersTo (Java 16)

```
void processWeakReference(WeakReference<MyObject> ref, MyObject obj) {

  if (ref.get() == obj) {...}          ✗

  if (ref.refersTo(obj)) {...}         ✓
}
```

Process + Reader = ♡ (Java 16)

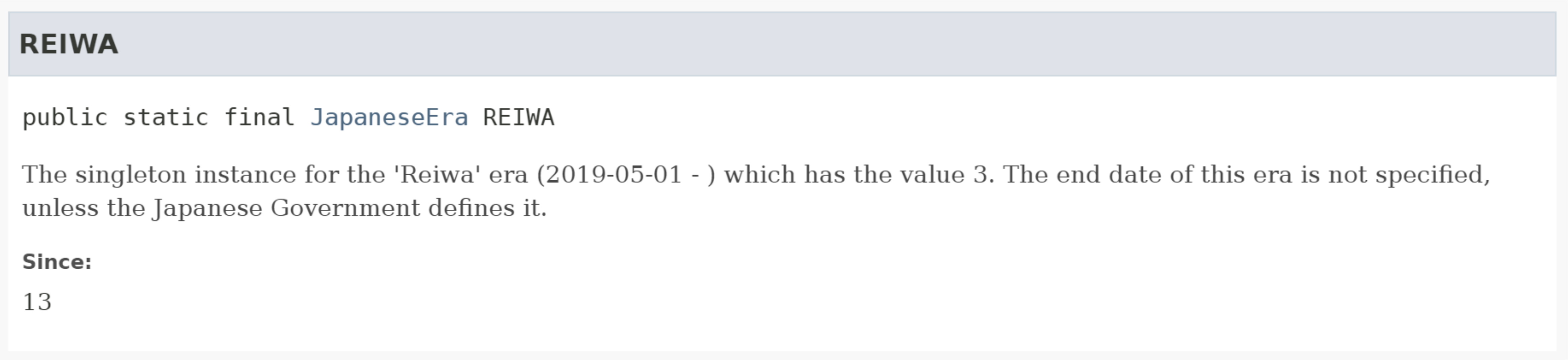
Было

```
InputStream errorStream = process.getErrorStream();
InputStream inputStream = process.getInputStream();
OutputStream outputStream = process.getOutputStream();
```

Стало

```
BufferedReader errorReader = process.errorReader();
BufferedReader inputReader = process.inputReader();
BufferedWriter outputWriter = process.outputWriter();
```

Class JapaneseEra



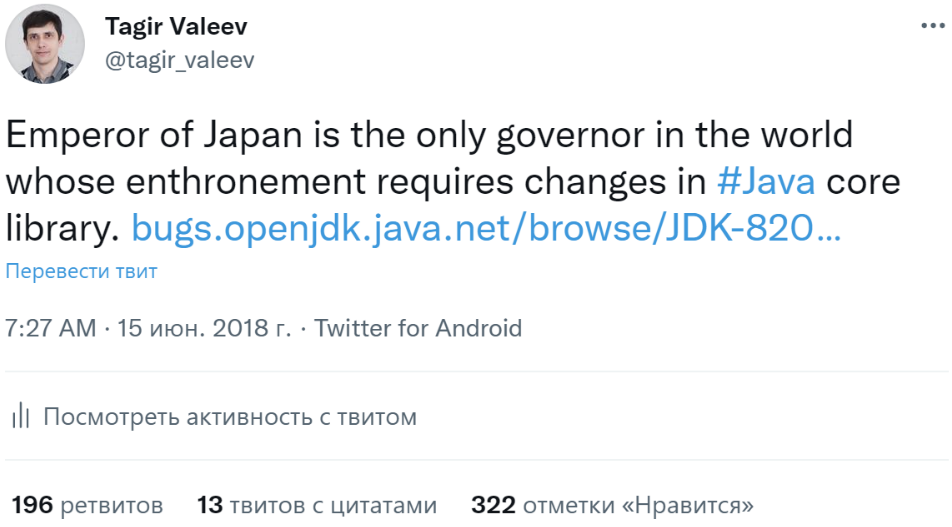
REIWA

```
public static final JapaneseEra REIWA
```

The singleton instance for the 'Reiwa' era (2019-05-01 -) which has the value 3. The end date of this era is not specified, unless the Japanese Government defines it.

Since:

13



Tagir Valeev
@tagir_valeev

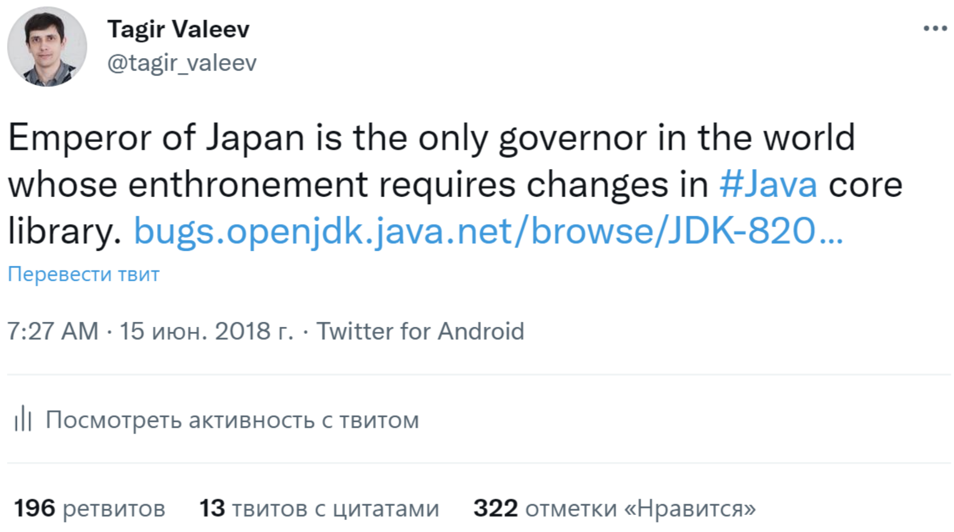
Emperor of Japan is the only governor in the world whose enthronement requires changes in #Java core library. bugs.openjdk.java.net/browse/JDK-820…

Перевести твит

7:27 AM · 15 июн. 2018 г. · Twitter for Android

Посмотреть активность с твитом

196 ретвитов **13** твитов с цитатами **322** отметки «Нравится»

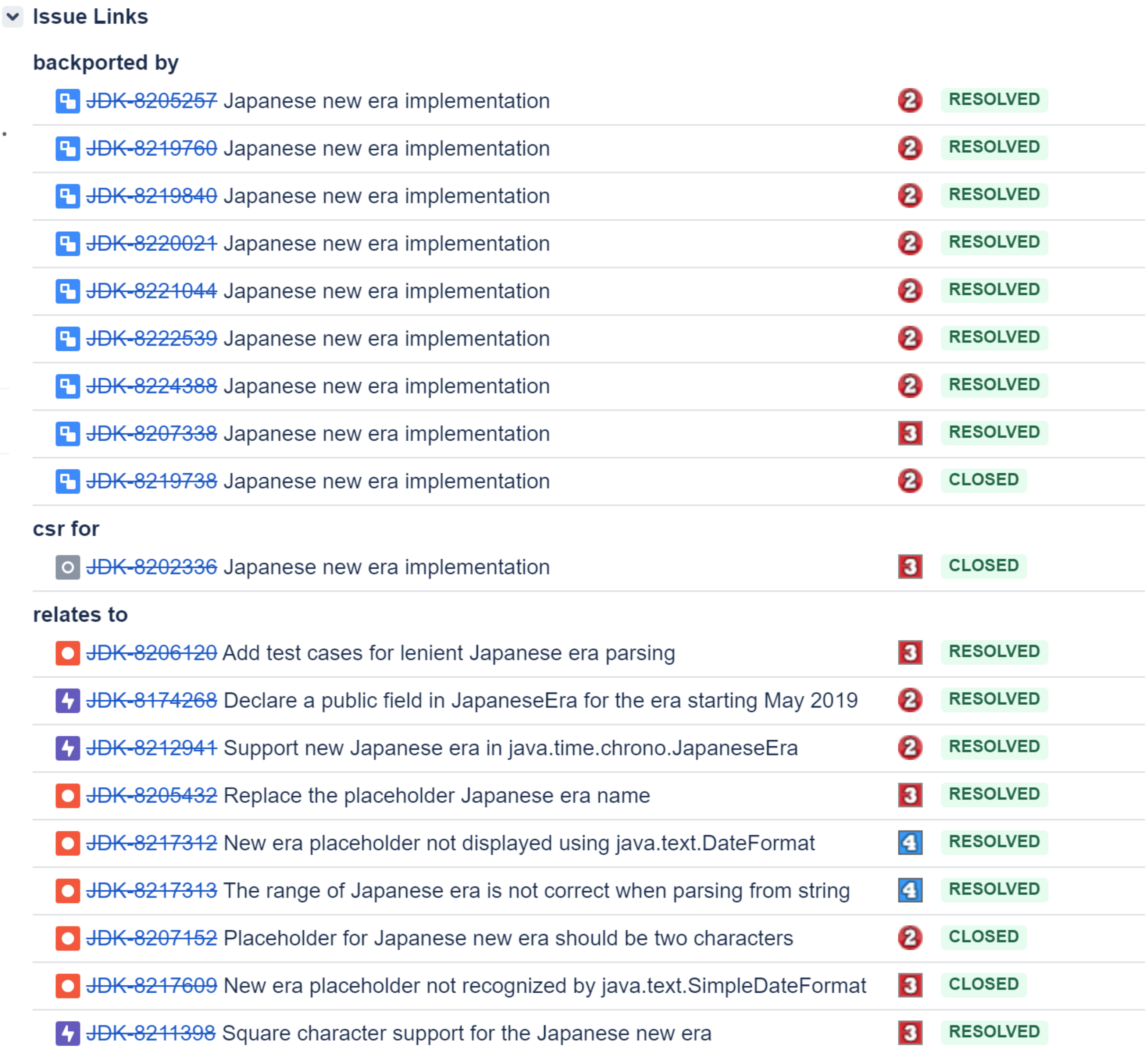


Tagir Valeev
@tagir_valeev

Emperor of Japan is the only governor in the world whose enthronement requires changes in #Java core library. bugs.openjdk.java.net/browse/JDK-820…

7:27 AM · 15 июн. 2018 г. · Twitter for Android

196 ретвитов 13 твитов с цитатами 322 отметки «Нравится»



Issue Links

backported by

~~JDK-8205257~~ Japanese new era implementation	2	RESOLVED
~~JDK-8219760~~ Japanese new era implementation	2	RESOLVED
~~JDK-8219840~~ Japanese new era implementation	2	RESOLVED
~~JDK-8220021~~ Japanese new era implementation	2	RESOLVED
~~JDK-8221044~~ Japanese new era implementation	2	RESOLVED
~~JDK-8222539~~ Japanese new era implementation	2	RESOLVED
~~JDK-8224388~~ Japanese new era implementation	2	RESOLVED
~~JDK-8207338~~ Japanese new era implementation	3	RESOLVED
~~JDK-8219738~~ Japanese new era implementation	2	CLOSED

csr for

~~JDK-8202336~~ Japanese new era implementation	3	CLOSED

relates to

~~JDK-8206120~~ Add test cases for lenient Japanese era parsing	3	RESOLVED
~~JDK-8174268~~ Declare a public field in JapaneseEra for the era starting May 2019	2	RESOLVED
~~JDK-8212941~~ Support new Japanese era in java.time.chrono.JapaneseEra	2	RESOLVED
~~JDK-8205432~~ Replace the placeholder Japanese era name	3	RESOLVED
~~JDK-8217312~~ New era placeholder not displayed using java.text.DateFormat	4	RESOLVED
~~JDK-8217313~~ The range of Japanese era is not correct when parsing from string	4	RESOLVED
~~JDK-8207152~~ Placeholder for Japanese new era should be two characters	2	CLOSED
~~JDK-8217609~~ New era placeholder not recognized by java.text.SimpleDateFormat	3	CLOSED
~~JDK-8211398~~ Square character support for the Japanese new era	3	RESOLVED

Спасибо за внимание

https://twitter.com/tagir_valeev
https://habrahabr.ru/users/lany
https://github.com/amaembo
tagir.valeev@jetbrains.com